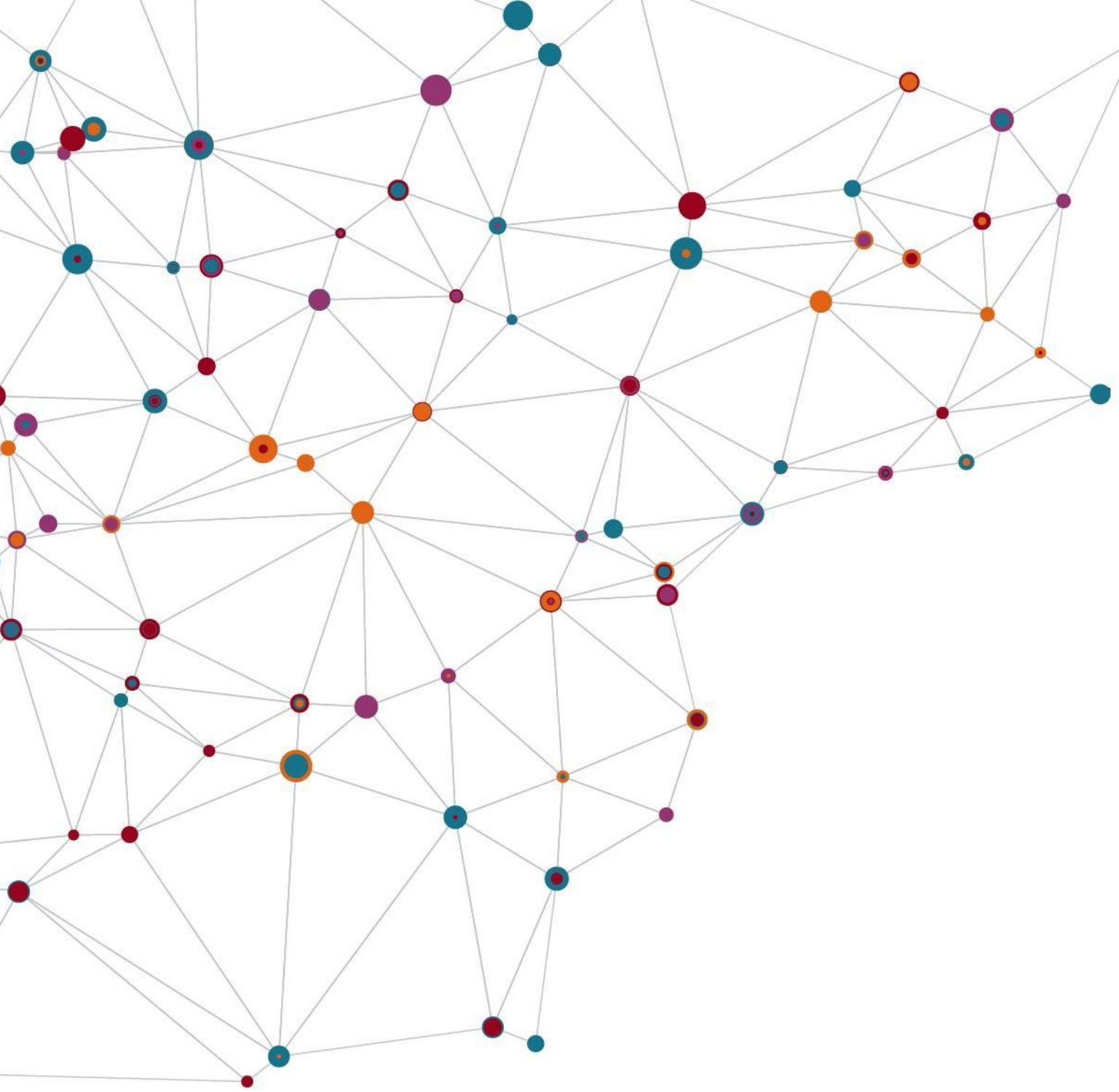


Formación Profesional

Unidad N°0 y 1

Clase 1: Presentación del curso.

¡Bienvenido/a a este curso denominado **"Introducción a la Programación de Videojuegos con Python y Godot Engine"**!

A continuación te contaremos las principales características del mismo y algunas recomendaciones para realizarlo.

Para más detalles te invitamos a leer el **Programa del curso**.

Objetivos.

- Los estudiantes deberán poder reconocer los recursos teóricos necesarios.
- Los estudiantes deberán formar el razonamiento para la resolución de distintos problemas que nos presenta la elaboración de un proyecto de videojuego, aplicando la solución más creativa y adecuada al proyecto.
- Los estudiantes deberán adquirir habilidades en la resolución de los problemas de programación.
- Los estudiantes deberán poder aplicar conceptos en programación estructurada en el desarrollo, resolver problemas lógicos de baja y mediana complejidad, dentro y fuera de un motor gráfico para desarrollo de videojuegos.
- Los estudiantes deberán poder publicar un juego en una plataforma online.

Contenidos.

Unidad	Clase	Título
0 y 1 Presentación. Diagramas de flujo. Pseudocódigo.	Clase 1	- Presentación / INTRODUCCIÓN. - ¿Qué es programar? ¿Qué es un Algoritmo? - Diagrama de flujo. Ejercicios online con Creately. - Pseudocódigo. PSeInt. Instalación. Ejercicios online con PSeInt.
2 Python.	Clase 2	- Introducción Python. Instalación. Ejercicios online con Python: Calculadora que suma y resta. - Procesos en Python: If/else, for, while. Ejercicios online con Python: Juego "Adivina el número". - Funciones en Python. Ejercicios online con Python: Mejorando la calculadora que suma y resta.
3 Diseño de VVJJ.	Clase 3	- Introducción al Diseño de VJ: MDA. Recursos online (assets). Documentación. GDD. TDD. EHC. Guión técnico. Backlog. Cronograma. Diagrama de gannt. Herramientas online de trabajo: Drive, Trello, Figma, Git, Itch.io. Play Store. App Store. - Análisis de videojuegos: Curva de dificultad. Curva de aprendizaje. - Ejercicios online: Analizando VVJJ. Creación de niveles con Spelunky/Kodu/Mario/etc.

4 Introducción a Godot. Estructura de trabajo en Godot. Replicando VVJJ en Godot. Recursos-Assets. SC-GUI-HUD.	Clase 4	- Introducción a Godot. Instalación. Interfaz de trabajo. - Escenas y nodos. Inspector. Creación de roca. Splash Screen. GUI. HUD. - Prácticas en Godot: Captura la bandera/Spaceship/Runner. - Creación de splash screen y un menú simple.
5 Producción y desarrollo de un VVJJ.	Clase 5	- Desarrollo de GDD / Actividad final / Programación de Juego.
	Clase 6	- Desarrollo de GDD / Actividad final / Programación de Juego.
	Clase 7	- Desarrollo de GDD / Actividad final / Programación de Juego.
	Clase 8	- Desarrollo de GDD / Actividad final / Programación de Juego.
6 Presentación. Publicación. Cierre.	Clase 9	- Actividad final / Programación de Juego / Pre-entrega.
	Clase 10	- Presentación de proyectos. Cierre / Publicación.

Orientaciones para el cursado.

El curso presenta una serie de unidades, que se irán haciendo visibles a medida que avances en las mismas.

Cada unidad presenta:

- Un desarrollo, donde a través de clases, textos, videos, esquemas, interactivos, links a webs, artículos, etc. podrás conocer los aspectos centrales del tema.
- Un cuestionario de autoevaluación de lo aprendido y ejercitación offline que al realizar exitosamente podrás pasar a la siguiente unidad.

Cerca del final del curso encontrarás una actividad integradora final, una actividad de elaboración personal cuya aprobación te permitirá certificar el curso.

Se estima una dedicación personal total de 40 hs. De ese modo, para realizarlo exitosamente en el término de diez semanas, se requiere dedicar, aproximadamente,4 horas por semana a las lecturas y realización de actividades.

Carga horaria.

Unidad	Clase	Horas de dedicación aprox.
0 y 1 Presentación. Diagramas de flujo. Pseudocódigo.	Clase 1	2 hs. Sincrónicas. 2 hs. Asincrónicas.
2 Python.	Clase 2	2 hs. Sincrónicas. 2 hs. Asincrónicas.

3 Diseño de VVJJ.	Clase 3	2 hs. Sincrónicas. 2 hs. Asincrónicas.
4 Introducción a Godot. Estructura de trabajo en Godot. Replicando VVJJ en Godot. Recursos-Assets. SC-GUI-HUD.	Clase 4	2 hs. Sincrónicas. 2 hs. Asincrónicas.
5 Producción y desarrollo de un VVJJ.	Clase 5	2 hs. Sincrónicas. 2 hs. Asincrónicas.
	Clase 6	2 hs. Sincrónicas. 2 hs. Asincrónicas.
	Clase 7	2 hs. Sincrónicas. 2 hs. Asincrónicas.
	Clase 8	2 hs. Sincrónicas. 2 hs. Asincrónicas.
6 Presentación. Publicación. Cierre.	Clase 9	2 hs. Sincrónicas. 2 hs. Asincrónicas.
	Clase 10	2 hs. Sincrónicas. 2 hs. Asincrónicas.
		40

Evaluación y acreditación.

Este curso presentará una actividad final que derivará, según nivel de desempeño, en la aprobación del curso.

La realización de las actividades intermedias son obligatorias, pero su calificación no pesará ni incidirá en la aprobación final.

Cada unidad del curso tiene su importancia, por lo que se propone un recorrido por todas ellas. Ese recorrido implica realizar las lecturas ofrecidas en las unidades y realizar la actividad de autoevaluación de cada una de ellas.

La actividad final de aplicación, que integra lo visto a lo largo del curso, deberá ser una producción original y propia, de realización individual.

Una vez aprobada la tarea final, siendo éste el único requisito de aprobación, se otorgará el certificado correspondiente.

Acompañamiento.

Durante el trayecto contarás con el apoyo de un servicio tutorial que responderá todas tus dudas, pero también con espacios de comunicación con todos los cursantes que estén realizando el curso, de modo que puedas compartir con ellos no solo tus problemáticas, si no también colaborar con otros.

Si bien el curso será de realización personal e individual no estarás solo/a durante ese tiempo.

Clase 1: Diagramas de flujo.Pseudocódigo.

Temas.

¿Qué es programar? ¿Qué es un Algoritmo? Diagrama de flujo. Pseudocódigo. PSeInt. Instalación.

Objetivos.

- Empezar a comprender conceptos básicos de programación.
- Poder describir un proceso mediante diagrama de flujo.
- Poder diferenciar código de pseudocódigo.
- Poder describir un algoritmo.

1. Desarrollo.

¿Qué es programar?

Si buscamos una descripción en pocas palabras sobre qué es programar, encontraríamos frases como: «crear software usando un lenguaje de programación», «darle instrucciones al ordenador» o «enseñarle al ordenador a hacer algo».

Este es un curso práctico y creemos que es mejor que vayas descubriendo en qué consiste programar, realizando precisamente esa actividad. En esta sección de introducción hablaremos de forma breve sobre algunos conceptos esenciales, algo así como el «abc» que nos permita comenzar a andar.

Algoritmos, programas y lenguajes de programación.

Para ayudar a entender la programación a un nivel básico se suele utilizar símiles, como las instrucciones de montaje de un mueble o una receta de cocina. En ellas explicamos cómo realizar algo a través de una serie de pasos detallados. Por ejemplo, al escribir una receta, primero hemos tenido que descomponer mentalmente el proceso de cocinar un plato en una serie de tareas con un orden lógico:

- Limpiar el pescado
- Echarle dos pizcas de sal
- Picar 20 gr. de cebolla
- Calentar 2 cucharadas de aceite en una sartén
- Dorar la cebolla
- etc…

Luego escribiremos esos pasos. Podría ser en español, en inglés o cualquier otro idioma, pero las instrucciones seguirán siendo las mismas.

Ahora bien, a este desglose de un proceso en pasos detallados y ordenados le denominamos **algoritmo** y al software donde transcribiremos estas órdenes usando

un lenguaje de programación concreto (Javascript, PHP, Python, Java …) para que pueda ser ejecutado por un ordenador, lo llamaremos programa.

La sintaxis de estos lenguajes de programación es bastante más simple que nuestros idiomas, además se utiliza un vocabulario y un conjunto de reglas mucho más reducido. Eso sí, son muy estrictas y debemos seguirlas para que el ordenador pueda interpretarlas sin que se produzca un error.

En resumen, estos programas son un conjunto de sentencias escritas en un lenguaje de programación que le dicen al ordenador qué tareas debe realizar y en qué orden, a través de una serie de instrucciones que detallan completamente ese proceso sin ambigüedad.

Diagramas de flujo.

Un diagrama de flujo es un diagrama que describe un proceso, sistema o algoritmo informático. Se usan ampliamente en numerosos campos para documentar, estudiar, planificar, mejorar y comunicar procesos que suelen ser complejos en diagramas claros y fáciles de comprender. Los diagramas de flujo emplean rectángulos, óvalos, diamantes y otras numerosas figuras para definir el tipo de paso, junto con flechas conectoras que establecen el flujo y la secuencia. Pueden variar desde diagramas simples y dibujados a mano hasta diagramas exhaustivos creados por computadora que describen múltiples pasos y rutas. Si tomamos en cuenta todas las diversas figuras de los diagramas de flujo, son uno de los diagramas más comunes del mundo, usados por personas con y sin conocimiento técnico en una variedad de campos.

Teniendo como punto de partida una situación cotidiana, como la de cocinar fideos, ¿cómo armarían un algoritmo y su diagrama de flujo? Los invito a pensar en esto y experimentar el diseño de un diagrama de flujo desde Creately, un editor de plantillas de diseño online que permite crear y editar diagramas de flujo, entre muchas otras cosas.

Ventajas y desventajas del pseudocódigo.

- Los diagramas de flujo ayuda a la comprensión del proceso al mostrarlo con un dibujo. El cerebro humano reconoce fácilmente los dibujos.
- Nos permite identificar los errores y nos da la oportunidad de agregarlo y mejorar el proceso.
- Es fácil identificar los procesos.
- Los diagramas complejos pueden ser muy laboriosos durante la planeación y el diseño del mismo.
- Puede ser difícil el seguimiento si el diagrama tiene diferentes caminos.
- No tiene normas fijas para la elaboración de los diagramas de flujos.

Componentes y sintaxis de un diagrama de flujo.

Una vez que tenemos nuestro diagrama de flujo solo tendremos que conocer las órdenes del lenguaje que realizan esas tareas que se especifican en el

diagrama, y que en este está dado por símbolos. Todos los símbolos han de estar conectados. A un símbolo de proceso pueden llegarle varias líneas. A un símbolo de decisión pueden llegarle varias líneas, pero sólo saldrán dos (Si o No, Verdadero o Falso). A un símbolo de inicio nunca le llegan líneas. De un símbolo de fin no parte ninguna línea.

Los símbolos que se usan para realizar los diagramas de flujo son lo siguientes:

En el Símbolo de decisión puede tomar los valores de salida SI o NO o también VERDADERO o FALSO.

El símbolo de Inicio o Final del diagrama puede ser un cuadrado con los bordes redondeados o una elipse.

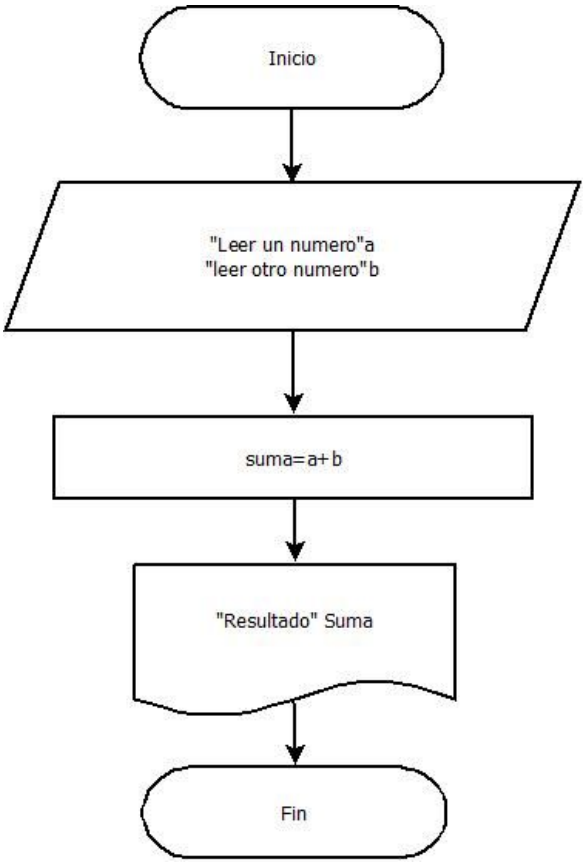
Se pueden utilizar colores para los símbolos.

Estructura de un algoritmo en diagrama de flujo.

Veamos un primer ejemplo muy sencillo.

Queremos hacer un programa informático que nos sume dos números y nos dé el resultado en pantalla.

Solución del ejemplo:



El símbolo de resultado es un símbolo usado en los diagramas para soluciones con el ordenador.

Es el símbolo de salida del resultado por la pantalla del ordenador.

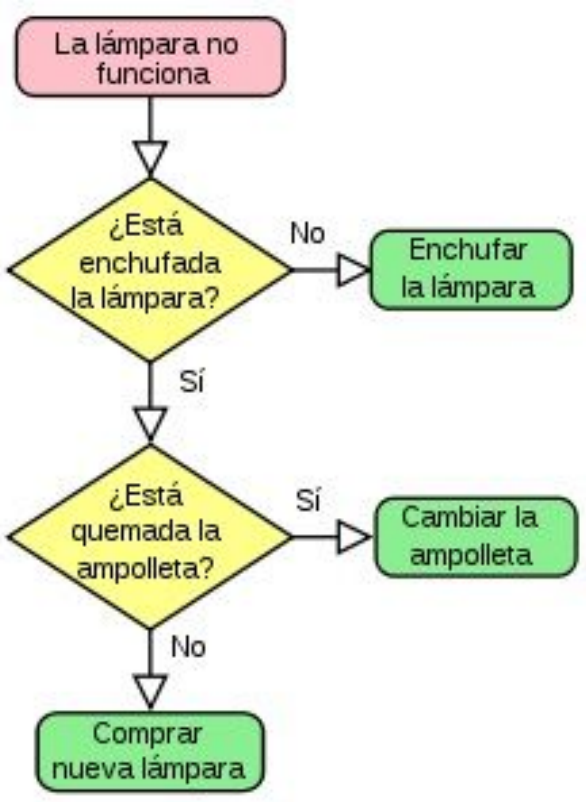
Ves que es muy sencillo, hay que ir poniendo los pasos lógicos que se deben seguir para realizar la tarea o el programa.

En el ejercicio tenemos el inicio y el fin, una entrada de datos, para meter los 2 números, una operación a realizar, la suma, y un resultado a mostrar.

Cada uno de esos pasos con su símbolo correspondiente en el diagrama.

Nuestro primer Diagrama de flujo.

Hagamos un diagrama de flujo para una operación sencilla. Imaginemos que tenemos una lámpara y queremos hacer el diagrama de flujo para saber qué hacer cuando la lámpara no funciona.



Operadores lógicos, aritméticos y relacionales.

Como su nombre lo describe, los operadores son expresiones que nos permiten indicar qué operación debe llevarse a cabo dentro de un proceso. Los hay de tres tipos.

ARITMÉTICOS: Los operadores aritméticos nos permiten, básicamente, hacer cualquier operación aritmética que necesitamos. Una suma, una resta, una multiplicación, etc.

RELACIONALES: Son operadores que se encargan de unir y comparar dos o más valores, se utilizan en comparaciones de parejas por los símbolos. >, <, >=, <=.

LÓGICOS: Son operadores de unión, también llamados compuertas lógicas, estos operadores pueden unir dos o más pares de valores comparados por medio de los operadores relaciones. &&, operador AND (Y) todas las condiciones deben ser verdaderas para que se ejecute una acción.

OPERADOR	SIGNIFICADO	EJEMPLO
ARITMÉTICOS		
+	Suma	1 + 2
-	Resta	2 - 1
+	Multiplicación	2 * 2
/	División	2 / 2

Mod %	Módulo o resto de la división por un entero	2 % 2
RELACIONALES		
>	Mayor que	3 > 2
<	Menor que	2 < 3
=	Asignación de valor	num = 1
==	Igual a	1 == 1
>=	Mayor o igual que	3 >= 3
<=	Menor o igual que	2 <= 3
LÓGICOS		
& Y	Conjunción	(7>4) & (2<1) Falso
| O	Disyunción	(7>4) & (2<1) Verdadero
! NO	Negación	! (2<5) Falso

Pseudocódigo.

El pseudocódigo es una forma de expresar los distintos pasos que va a realizar un programa, de la forma más parecida a un lenguaje de programación. Su principal función es la de representar por pasos la solución a un problema o algoritmo, de la forma más detallada posible, utilizando un lenguaje cercano al de programación. El pseudocódigo no puede ejecutarse en un ordenador ya que entonces dejaría de ser pseudocódigo, como su propio nombre indica, se trata de un código falso (pseudo = falso), es un código escrito para que lo entienda el ser humano y no la máquina.

Aprender a escribir pseudocódigo para la resolución de un problema permite hacer mucho más sencilla su programación en un lenguaje convencional, por lo que antes de empezar a estudiar un lenguaje de programación, sería conveniente realizar una introducción que nos muestre el ciclo de desarrollo de un programa mediante pseudocódigo.

Podemos considerar al pseudocódigo como un lenguaje intermedio, que se encuentra en medio de nuestro propio lenguaje y el lenguaje de programación que entiende el ordenador.

Ventajas y desventajas del pseudocódigo.

Las tareas más complejas o repetitivas pueden representarse de forma más sencilla ya que está escrito en un lenguaje sencillo y no estructurado que permite una transición sencilla al lenguaje de programación, más complejo y estructurado. Tener un programa escrito en pseudocódigo facilita la tarea de programar en un lenguaje formal y mejora la calidad en la resolución de problemas, además de reducir el espacio necesario a la hora de desarrollar un problema.

El pseudocódigo llega donde el diagrama de flujo no lo hace. La solución de un diagrama de flujo suele ser la ideal, pero no suele ser fácil de implementar al crear el programa. El pseudocódigo permite que el diseño del programa y su implementación sean muy parecidos.

La curva de aprendizaje del pseudocódigo es baja por lo que facilitan enormemente el aprendizaje de la programación y la iniciación a lenguajes de programación más avanzados y complejos. Por lo tanto, se trata de una herramienta educativa interesante.

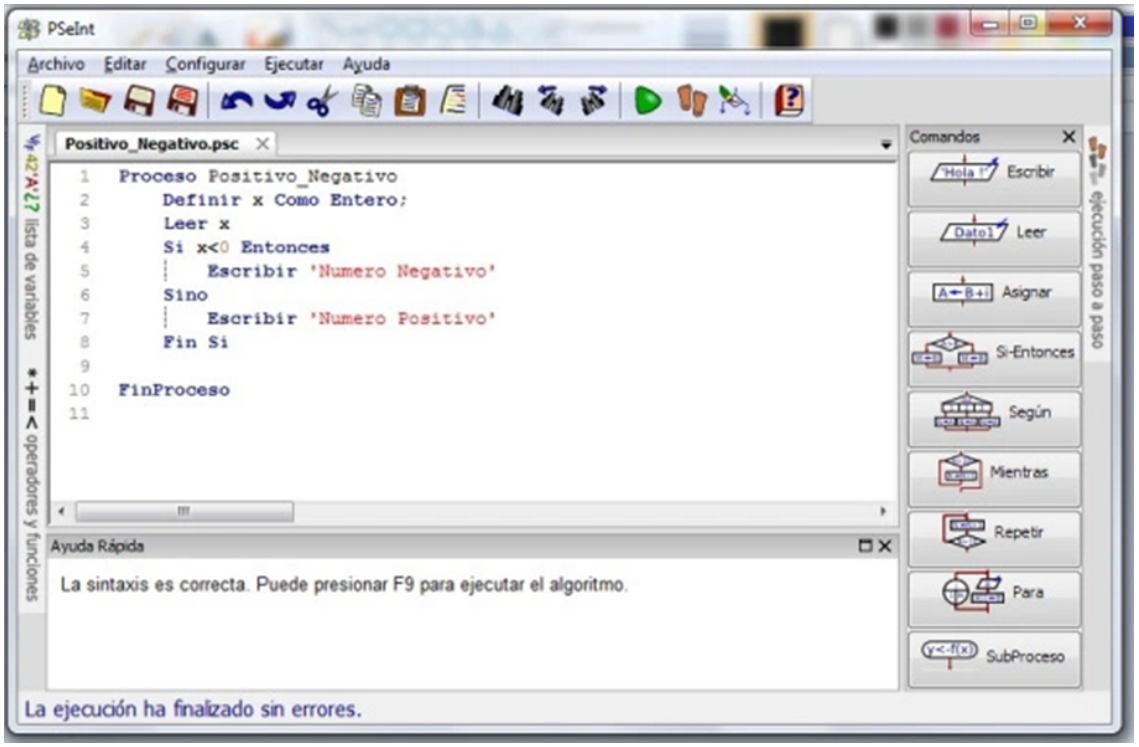
El pseudocódigo, al ser independiente del lenguaje de programación, permite que su uso se pueda aplicar utilizando diferentes lenguajes y permitiendo que el programador no tenga que ser la misma persona que escribió el pseudocódigo.

Una de las desventajas del uso de pseudocódigo es la falta de normas, que puede hacer que la lógica de un programa, resulte complicada de ver por el programador que va a implementar este pseudocódigo. Además, en el caso de problemas muy extensos, puede llegar a ser difícil de entender.

Componentes y sintaxis del pseudocódigo.

Para escribir programas utilizando pseudocódigo es necesario seguir reglas y sintaxis específicas para que puedan ser leídas y comprendidas por cualquier programador. Es muy útil utilizar herramientas que faciliten esta escritura de pseudocódigo, como es el caso de PSeInt, que asiste con un simple e intuitivo pseudolenguaje en español y que además incluye un editor de diagramas de flujo.

Un programa escrito en pseudocódigo debe permitir escribir acciones sencillas, de proceso, de control, de descripción y compuestas.



Estructura de un algoritmo en pseudocódigo.

Todo algoritmo en pseudocódigo tiene la siguiente estructura general:

```
Proceso SinTitulo
    acción 1;
    acción 2;
    ...
    acción n;
FinProceso
```

- Comienza con la palabra clave Proceso (o alternativamente Algoritmo, son sinónimos) seguida del nombre del programa.
- Le sigue una secuencia (Estructura de control secuencial) de instrucciones. Una secuencia de instrucciones es una lista de una o más instrucciones y/o estructuras de control.
- Finaliza con la palabra FinProceso (o FinAlgoritmo).
- No se diferencia entre mayúsculas y minúsculas. Son preferibles las minúsculas, aunque a veces se añaden automáticamente los nombres con la primera letra en mayúsculas.

Comentarios.

Los comentarios son datos que los programadores dejan para guiar a la persona que lee el código. Se pueden introducir comentarios luego de una instrucción, o en líneas separadas, mediante el uso de la doble barra en Pseint (//). Todo lo que precede a //, hasta el fin de la línea, no será tomado en cuenta al interpretar el algoritmo.

Nuestro primer programa.

Vamos a escribir nuestro primer programa llamado “Hola Mundo”. En este primer programa, le pediremos al ordenador que escriba y muestre en pantalla el saludo “Hola Mundo”. Para ello, seguiremos los siguientes pasos:

- En Proceso, agregaremos el nombre del programa (Hola Mundo).
- Luego escribiremos la orden ESCRIBIR “...”, la cual le indica al ordenador que muestre en pantalla todo lo escrito entre las comillas.
- Siempre al finalizar el programa debemos escribir FinProceso para indicar la finalización de las acciones.

El pseudocódigo podría ser:

```
Proceso Hola_Mundo
    Escribir "Hola Mundo";
FinProceso
```

Si quisiera dejar comentarios en el pseudocódigo para que el programador modifique este programa, el pseudocódigo podría ser:

```
Proceso Hola_Mundo
// ESCRIBIR:  Esta herramienta permite al programador mostrar texto en pantalla.
// LEER : Permite capturar por teclado los datos que el programador ingrese.
// ASIGNAR: Permite asignar a una variable una operación matemática o igualarla a otra
    Escribir "Hola Mundo";
FinProceso
```

2. Autoevaluación

Descripción.

Te invitamos a que evalúes tu comprensión y dominio del tema tratado. Al superar el desafío se te habilitará la siguiente unidad y podrás continuar.

Instrucciones.

A continuación hallarás algunas preguntas que pondrán a prueba tus saberes. Para pasar esta etapa es importante que hayas leído la unidad y practicado con los ejercicios que propuso la misma.

Preguntas y respuestas.

Elegir la respuesta correcta:

1. ¿Qué es un diagrama de flujo?

a) Un conjunto de instrucciones o reglas bien definidas, ordenadas y finitas que permiten realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad.

b) Es un diagrama que describe un proceso, sistema o algoritmo informático.

c) Es una relación de variables que pueden ser cuantificadas para calcular el valor de otras de muy difícil o imposible cálculo y que suministra una solución para un problema

d) Las anteriores respuestas no son correctas

2. ¿Cómo se define a los pasos de un algoritmo que se ven reflejados en una estructura de gráficos?

a) Programación

b) Asignación de datos

c) Diagrama de flujo.

3. En un diagrama de flujo, ¿cómo se le indica al sistema que dé un resultado?

a) Con la orden ESCRIBIR

b) Con la orden LEER

c) Con el símbolo ELIPSE

4. ¿Qué es un algoritmo?

a) Un conjunto de instrucciones o reglas bien definidas, ordenadas y finitas que permiten realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad.

b) Es una igualdad entre dos expresiones algebraicas, denominadas miembros, en las que aparecen valores conocidos o datos, y desconocidos o incógnitas, relacionados mediante operaciones.

c) Es una relación de variables que pueden ser cuantificadas para calcular el valor de otras de muy difícil o imposible cálculo y que suministra una solución para un problema.

5. En Pseint, ¿cómo se le indica al ordenador que muestre un texto en pantalla?

a) Con la orden ESCRIBIR.

b) Con la orden LEER.

c) Con el símbolo ELIPSE.

DEVOLUCIÓN:

Buen trabajo. Ya hemos experimentado con algoritmos, diagramas de flujo y pseudocódigo. Ahora es tiempo de pasar a realizar algunos ejercicios más para luego avanzar a la utilización de uno de los lenguajes de programación más utilizados hoy en día, Python. Nos vemos en la próxima clase.

3. Tarea.

1. ***Diseña un diagráma de flujo que nos muestre el resultado del área de un triángulo en pantalla.***
2. ***Diseña un diagráma de flujo que nos diga si un número es par o impar.***
3. ***Diseña un diagráma de flujo que nos muestre la suma de los 50 primeros números en pantalla.***
4. ***Diseña un algoritmo en pseudocódigo que nos muestre el resultado del área de un triángulo en pantalla.***
5. ***Diseña un algoritmo en pseudocódigo que nos diga si un número es par o impar.***
6. ***Diseña un algoritmo en pseudocódigo que nos muestre la suma de los 50 primeros números en pantalla.***