VIDEOJUEGOS
INTRODUCCIÓN A PYTHON

CLASE 1

Contenido

“

Este espacio curricular permite fortalecer el espíritu crítico y la actitud creativa del futuro graduado. Lo alienta a integrar los conocimientos previos, recreando la práctica en el campo real, al desarrollar un proyecto propio.

”

PRORAMACIÓN I

Nos detenemos un momento en este lugar, no quisiera avanzar sin darles un concejo. Programar es una de las actividades más emocionantes que pueden vivir, el placer que se obtiene al ver el producto terminado es único. Pero los comienzos son algo turbulentos, y suele cuásar algo de estrés.

Los primeros pasos son los más importantes, pero también los más difíciles. Pueden tener algunas frustraciones, pero a no abandonar. La curva de aprendizaje al principio puede exigir mucha dedicación y esfuerzo. Van a sentir que dedican muchas horas de estudio y avanzan poco, pero es normal. Créanme que, en algún momento, esa ecuación cambia y se invierte por completo.

Ese punto de inflación llega, se los puedo asegurar. ¿Pero cuando? Bueno eso depende de Uds. y de las horas que les dediquen a sus estudios. Por eso digo que programar es un ejercicio de paciencia, es tiempo, es estudio… pero al final, los frutos lo valen…

¡¡Así que... A disfrutar... Que, al terminar el cuatrimestre, ¡¡estaremos jugando con nuestros propios juegos!!

Presentación

Es esta asignatura, vamos a incursionar en el desarrollo de videojuegos 2D usando Python como lenguaje de programación. El desarrollo involucra competencias no solo de programación, también debemos tener capacidades en el modelo de niveles, uso de Sprite entre otras cosas.

El ámbito del desarrollo de videojuegos está dominado por motores como Unity3D o Unreal Engine. Estos motores promocionan el uso de C3 y c++ como lenguajes de programación. De todos modos, no son los únicos caminos para desarrollar un videojuego. No usarlos nos hace el recorrido un poco más complejo, pero podemos crear proyectos interesantes usando otras plataformas. En este caso, vamos a trabajar con Python y sus librerías.

Nos detenemos un momento en este lugar, no quisiera avanzar sin darles un concejo. Programar es una de las actividades más emocionantes que pueden vivir, el placer que se obtiene al ver el producto terminado es único. Pero los comienzos son algo turbulentos, y suele cuásar algo de estrés.

Los primeros pasos son los más importantes, pero también los más difíciles. Pueden tener algunas frustraciones, pero a no abandonar. La curva de aprendizaje al principio puede exigir mucha dedicación y esfuerzo. Van a sentir que dedican muchas horas de estudio y avanzan poco, pero es normal. Créanme que, en algún momento, esa ecuación cambia y se invierte por completo.

Ese punto de inflación llega, se los puedo asegurar. ¿Pero cuando? Bueno eso depende de Uds. y de las horas que le dediquen a sus estudios. Por eso digo que programar es un ejercicio de paciencia, es tiempo, es estudio… pero al final, los frutos lo valen…

Así que... A disfrutar... Que, al terminar el cuatrimestre, ¡¡estaremos jugando con nuestros propios juegos!!

Contenidos mínimos

Programación I es una asignatura de la carrera de Videojuegos, vamos a dividir cada encuentro en Teoría y práctica. Durante el desarrollo nos adentraremos en distintos paradigmas de programación, pero eso lo veremos luego.

A modo de introducción Podemos resumir los contenidos mínimos de la asignatura en estos 4 focos. En todos los casos, el alcance de cada tema será presentado tanto en su teoría como su práctica. Y en pocas palabras podemos decir que:

Daremos una introducción a las estructuras básicas, que en general aplican a cualquier lenguaje. Pero en Python, algunas cuestiones relacionadas a su sintaxis lo hacen un poco más complejo al principio. Aquí, veremos cómo usar variables, cadenas, colecciones. Veremos cómo aprovechar las estructuras de decisión y los bucles de repetición. Terminaremos programando funciones y el uso de algunas librerías elementales de Python

como también los canales de entrada y salida que me brinda el lenguaje.

Es muy importante, tanto en Python como en cualquier otro lenguaje, trabajar en las competencias que nos permiten representar con un nivel adecuado de abstracción cualquier solución a los problemas planteados. Aunque no es un tema de la asignatura, trabajaremos indirectamente sobre algunos tópicos de algoritmia que pueden serles útiles en el futuro.

A lo largo de los años, se fueron gestando distintos modelos para desarrollar una aplicación digital. Podemos decir que un paradigma es un conjunto de acuerdos que guían al desarrollador al momento de sentarse a programar. El problema es que, como ocurre siempre, estos acuerdos suelen cambiar con el tiempo. En la actualidad, algunos paradigmas están arraigados en la comunidad que otros. Lo interesante de Python es que nos permite trabajar con varios de ellos.

algunos lenguajes, como C, brindan un conjunto limitado de funciones. Pero, permiten importar paquetes para potenciar el desarrollo. Python no es ajeno a ello, y en este espacio veremos varias de las librerías del lenguaje.

Criterios de aprobación

Tenía que llegar, hablemos de la parte fea de estudiar... ¿Cómo hacemos para aprobar?

Para aprobar, tendrás que sumar créditos en varios desafíos, en todos los casos la nota debe ser igual o superior a 6 (seis).

Desafíos Semanales: Cada clase, tendrás actividades para completar, muchas de ellas se autoevalúan. Pero son muy importante para que vos mismo puedas ver si la base de conocimiento esta firme.

Trabajos prácticos: Son tres en total, dos que sirven como instancia de autoevaluación de los temas vistos, más un trabajo final integrador donde tendrás que aplicar todo lo visto en la asignatura. El TP integrador, lo deberás mostrar antes de finalizar el cuatrimestre. Cada instancia de entrega tendrá también una defensa individual del trabajo y su contenido.

Exámenes: Tendremos dos instancias de evaluación, una al terminar el mes de mayo y la otra a fines de junio. Los temas que se observarán en cada instancia serán todos los temas vistos hasta el momento en cada una de las clases

Por último, el presentismo, recuerda que no puedes tener menos del 75% de asistencia.!! ¿Esto es lo más fácil, no me digas que no puedes cumplirlo? 😉

¡¡Si al terminar el cuatrimestre cumpliste y aprobaste todo esto, felicitaciones!! La materia **es tuya…**

Trabajos Prácticos

Te cuento como será el primer trabajo práctico. Se entrega a las 3 semanas de comenzar, pero es muy sencillo. Lo llamamos juego de letras, porque el objetivo es que desarrolles un juego que use letras como

elemento principal. Vamos a ir viendo en cada clase, los temas... ¡Pero terminarlo va a depender solo de vos!

El juego de letras es muy aplicable a un ámbito educativo, y encontrarás miles de ejemplos para analizar y programar. Lo importante, es que ya piensen como un futuro desarrollador de videojuegos y apliques todo lo que ves en otras asignaturas. ¡En programación I vemos sólo programación, pero recuerda que un juego no es solo código!

La originalidad será premiada, y todo lo que hagas para que el juego sea motivador por supuesto también tendrá su mérito y su premio. En estos juegos, a veces nos traiciona la haraganería, porque seguro ya habrá en la Internet varios ejemplos resueltos. No es malo ver el trabajo de otros. Pero si quieres aprender, mi recomendación es que programes tu propio juego.

Confío en tu buen juicio 😉

A esta altura, estamos con dos meses de experiencia. Por eso, el desafío es más interesante. Tal vez eres muy joven, pero búscalo... Carmen Sandiego fue de los juegos que más me enseñaron cuando era joven. Hoy tenemos una versión con un estilo moderno, pero busca información de la primera versión porque esa es la idea que te propongo como TP.

Vamos a desarrollar un juego narrativo, juego de aventura textual o conversacional. Son estilos de la década de los '70, pero muy aplicables en la actualidad. El formato es muy simple, toda la descripción de la situación proviene principalmente de un texto. Luego, el jugador debe ingresar algún dato y dependiendo de ese dato el camino del juego puede cambiar... Simple... ¿No?

A esta altura, ya estarás programando como el mejor, por lo cual es bueno que busques algún fin para el juego. Por ejemplo, enseñar historia, u ortografía... No sé... Deja volar tu imaginación

Trabajo Integrador, no hay mucho para explicar. ¡Usa todo el conocimiento que tienes en tu mochila para crear el mejor juego de tu vida! O por lo menos el mejor hasta ahora… ☺

Calendario

Te voy a dejar en el campus un calendario de clases y contenidos, es sólo una guía para ordenarnos. Todos sabemos que la vida no es tan estática, y por supuesto el calendario tampoco…

Día 1

1. Instalar Python
2. Instalar Vs Code
3. Agregar Extensiones

Entorno de Trabajo

Vamos a llamar entorno de trabajo al ecosistema de software que instalamos en nuestra PC. Al momento de tomar esta decisión, debemos pensar en que sistema operativo vamos a trabajar, y luego cual es el software que necesitamos.

En este caso, solo necesitamos un IDE (Integrated Development Environment). Son muchos los entornos posibles, te cuento a continuación algunos de ellos:

"IDE de 64 bits más rápido y confiable. Incluye .NET 6, incluida la compatibilidad con MAUI, las aplicaciones Blazor y Recarga activa. Finalizaciones mejoradas de IntelliCode. Cadena de herramientas más reciente para el destino en C++20 [1]". Tenemos varias versiones profesionales de pago y una versión gratuita llamada Visual Studio Community.

Xcode es una IDE muy potente para usar en OS Mac, deberás configurarlo, pero es muy práctico para desarrollar.

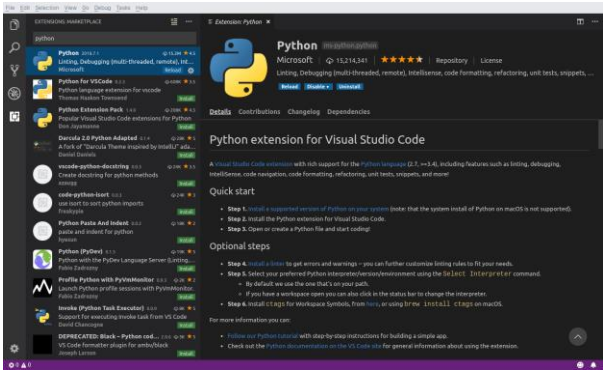
Pycharm es una alternativa interesante si no quieres usar un editor de Microsoft y no estar tan atado al sistema operativo. Tiene licencias educativas por lo que podrás probarlo. La empresa que lo desarrolla es JetBrain.

Otra alternativa a Visual Studio con una versión más liviana es VSCode[2]. Ofrece casi las mismas utilidades que su padre Visual Studio, tal vez te cueste un poco al principio si esas la versión profesional.

Ahora es cuando, saltamos la democracia. En esta edición vamos a trabajar con VsCode.

Instalación de Vscode

Lo primero que haremos es descargar el instalador, para eso entramos a la página https://code.visualstudio.com/ y buscamos el archivo para instalar en la versión de OS que queremos y en el idioma que más nos guste.



Instalación de Python

Algo importante que debemos saber, es que VSCode es un IDE multi uso. En el caso de Python, debemos descargar un intérprete.

Pasos para seguir:

[1] https://visualstudio.microsoft.com/es/

[2] https://code.visualstudio.com/

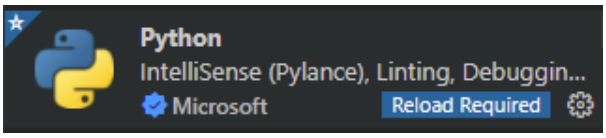
1- Descargar el intérprete de la página de descarga. Es importante descargar la versión 3 o superior. https://www.python.org/downloads/
2- Siga los pasos de instalación.

Agregar Extensiones

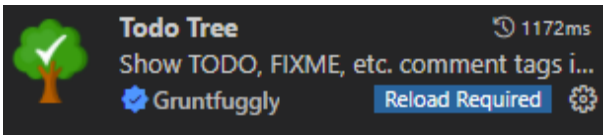
Para mejorar la experiencia de trabajo, vamos a instalar algunas extensiones. Vas a encontrar muchas extensiones, siempre es aconsejable instalar aquellas que tiene un autor conocido.

primero seleccionamos el icono de extensiones del editor

Luego en el buscador escribimos Python, cuando encontramos la extensión de Python la instalamos



Como adicional, te invito a instalar esta extensión de administración de comentarios que usaremos luego para mantenimiento del código.



Para ayudarte, te comparte este video https://youtu.be/Pcn8nUBJyPQ

Lenguaje de Programación

Si lo vemos desde un punto de visto mucho más amplio, Podemos decir que el lenguaje es un conjunto de símbolos escritos u orales por medio del cual dos personas pueden comunicarse.

¿Cuál es el sentido de un lenguaje?

Podemos decir que el fin de un lenguaje es permitir que dos personas puedan comprender un determinado tema, con el fin de comunicar o ponerse de acuerdo. Es claro, y creo que no abrimos debate con esto. Es necesario que ambas personas compartan o entiendan el mismo lenguaje, caso contrario la comunicación puede tornarse un poco más compleja. Y por ello, no poder compartir la información necesaria para poder resolver el problema o asunto que los convoca.

Ahora bien, esto se complica más para los desarrolladores. Porque no se trata de lograr que dos personas, con un promedio intelectual razonable, traten de comunicarse. En este caso, necesitamos comunicarnos con una computadora. Y claro, esto es mucho más difícil.

Es por ello, que tenemos que usar lo que llamamos un "lenguaje de programación". Son muchos lenguajes, pero en nuestro caso usaremos Python.

Clasificación de los lenguajes

Antes de comenzar con un lenguaje de programación, en este caso Python, tratemos de entender que es un lenguaje para la computadora. Cuando nace la computadora, esta solo entendía una cosa – "Encendido/apagado". Esto, se conoció luego como el código binario representado por un 1 (uno) y un 0 (cero). Sencillamente, decía cuando estaba encendido (tenía energía) y cuando estaba apagado.

Con el tiempo, representar datos con un 0 o un 1 no fue suficiente, la evolución de la tecnología demandada mucho más. Para afrontar esta necesidad, comenzaron a representar datos con grupos de más de un solo digito. Llegando así al Byte, este era un grupo de 8 bits. La combinación de 8 bits me permitía representar hasta 256 datos diferentes, que no era poca cosa.
Uds. se preguntarán, ¿se puede lograr mucho con un Byte? —Todos los caracteres que puedes ver en el teclado están representados en una tabla que se conoce como tabla ASCII, en total son 256; ¿esto te responde la pregunta?

Pues bien, no es intención de esta asignatura dar una clase de historia de la computación. Es por eso, que dejaremos la explicación de como llegamos del bit al teraByte 😉

En consecuencia, solo nos interesa saber que la computadora solo conoce el código binario. Entonces, ¿Dónde entra Python? Te lo explico a continuación.

Es raro que programemos en el lenguaje que la computadora entiende. Podrías, si lo quisieras, pero no te lo recomiendo.

Para evitar programar en un modo tan complejo, se crearon lenguajes de programación. Según el estilo de la sintaxis del lenguaje se puede clasificar en Bajos (cercanos a la PC), Medios y Altos (cercanos al humano). LA diferencia es que cuando más cerca estemos del nivel Alto, las palabras que formen parte del conjunto de comandos del lenguaje serán más claras para el humano y menos entendible para la computadora.

Clasificación

Hoy, programamos siempre en lenguajes de Alto nivel. Pero si un día quisieras programar directamente sobre una placa de video, tal vez tengas que usar lenguajes de bajo nivel. Entonces, ¿Cómo pasamos de un lenguaje de alto nivel a lenguaje que el pc entiende? Usando compiladores o intérpretes.

Lenguaje máquina

Como comentamos anteriormente, el leguaje de la computadora es el Código binario. Pero, cuando lo tenemos que representar muchas veces se muestra en código Hexadecimal, pero solo es una manera de interpretarlo para el humano.

Profesor: Pablo Vilaboa

Para resumir, lenguajes como Python, C# o java son considerados de alto nivel.

Lenguaje máquina

0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1

cuando el objetivo sea desarrollar videojuegos, usaremos lenguajes de alto nivel. estos, agrupan todos aquellos lenguajes que se aproximan más al lenguaje natural del programador. Su propósito radica en poder desarrollar sin un alto grado de dependencia del hardware que se utilice.

Como programamos alejados del lenguaje de la máquina, necesitamos un programa traductor que se encargue de convertir (compilar o interpretar) el código fuente con el lenguaje de alto nivel en un código que la computadora comprenda, es decir código máquina.

Lenguajes interpretados o Compilados

A esta altura ya sabemos que es un lenguaje de programación. Cuando creamos el archivo (con Python), lo que estamos creando es nuestro código Fuente. El código Fuente es la base de todo nuestro trabajo, y generalmente lo que más valor tiene para nosotros.

El código fuente sufre un proceso de transformación para traducirlo al lenguaje de la máquina, pero esto es cuando lo queremos ejecutar y hacerlo funcionar. Hasta allí, no vemos un problema latente. Pero, cuando ese mismo código lo tiene que ejecutar otra persona comienza la parte difícil. Una de las

razones, es que no quiero compartir mi código fuente, solo quiere compartir el resultado de mi código fuente. Cuando se quiere compartir el producto con otra persona, el código tiene que pasar por un proceso de traducción y además uno que me permita implementar de alguna manera ese producto. Generalmente, esta implementación se resuelve convirtiendo el código fuente un programa ejecutable.

Esto parece resolver el problema de compartir mi producto, si el código fuente lo convierto en un .EXE el código fuente se vuelve ilegible. El punto importante que tenemos que considerar es si, el lenguaje se solo interpretable o compilable. Para entender esto último, veamos una definición muy simple de cada concepto:

Interpretado: el proceso traducción a lenguaje máquina se hace cada vez que se ejecuta. Es decir, no tengo un .EXE

Compilado: El proceso de traducción se hace por única vez, luego de obtener el .EXE ese archivo se usan por el tiempo que uno desea.

Un ejemplo de esto es javascript, no es un lenguaje compilable esto significa que no obtengo un .EXE. Esto lo notaras al entrar en una página WEB, siempre veras la fuente de javascript y podrás tomar su código. Python, también es un lenguaje interpretado, pero más adelante vamos a ver cómo crear un .EXE

Ahora ya sabes cómo cuidar tu código fuente, podrás hacer tus juegos y compartir el .EXE a tus amigos.

Ese archivo con nuestro código tiene la solución al problema planteado. Esa solución, es decir la estrategia que programamos, se conoce como algoritmo.

¿Qué es un algoritmo?

Ya hemos cubierto las bases para entender porque necesitamos un lenguaje de programación y como puedes desarrollar estos programas. Pero, queda una parte que es la más importante.

¿Cómo pasamos la idea al código?

El lenguaje no es suficiente, para lograr explicarle a la computadora lo que queremos. Necesitamos seguir una lógica y respetarla. Cuando hablamos de algoritmos, decimos que es un conjunto ordenado de pasos que permiten obtener un resultado o encontrar la solución a un problema. Simplificando, decimos que, partiendo de un conjunto de datos de entrada, siguiendo pasos sucesivos no ambiguos se llega a un resultado final y se obtiene la solución al problema. Con el tiempo, vamos a estandarizar procesos algorítmicos para cada problema y la practica hará que encuentres esa solución mucho más rápido.

A lo largo de todo este cuatrimestre, vamos a ejercitar nuestras competencias para resolver por medio de algoritmos, todo tipo de situaciones.

Para decir que es un algoritmo, este código/solución debe respetar tres premisas simples:

1. **PRECISO** (Orden)
2. **DEFINIDO** (Resultado)
3. **FINITO**

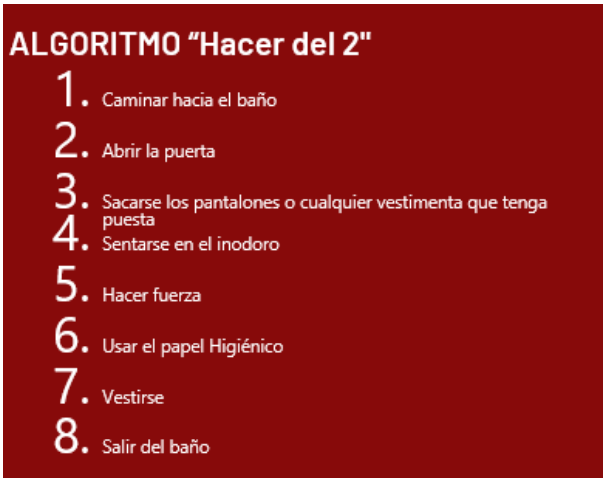
Cuando decimos que un algoritmo, es un conjunto de pasos no ambiguos, queremos decir que debe respetar por lo menos estas tres premisas

Que sea Preciso: Implica el orden de realización de cada uno de los pasos.

Que sea Definido: Si se sigue dos veces, se obtiene el mismo resultado.

Que sea Finito: Tiene un número determinado de pasos, implica que tiene un fin.

Veamos un ejemplo …



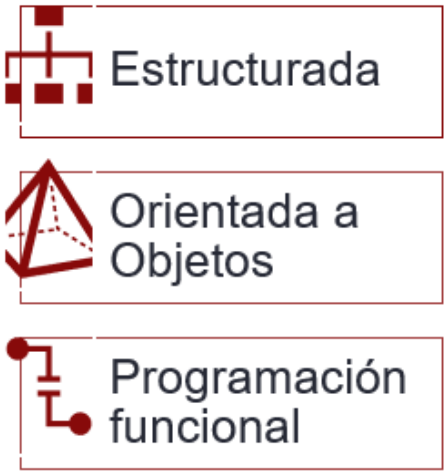
Ya hemos abordado bastante del tema, sabes que es un lenguaje y sabemos porque programamos en Python. Ahora bien, ese algoritmo que programamos, lo podemos encarar usando diferentes técnicas. Esas técnicas definen la manera que lo hacemos y se conocen como paradigmas de programación. Cada lenguaje, abona a un determinado paradigma.

Paradigmas de programación

Programación estructurada: También conocido como paradigma imperativo. Es un tipo de programación que se define mediante subrutinas y flujos de control

Programación Orientada a Objetos: En este paradigma se construyen modelos de objetos considerando un nivel de abstracción según el problema a resolver

Programación funcional: es un paradigma declarativo. En este paradigma las funciones tienen un rol protagónico, y podrán ser asignadas a variables y ser usadas como entradas y salidas de otras funciones



Introducción a Python.

Antes de hablar de Python, digamos que para ser un lenguaje de programación debe agrupar un conjunto de instrucciones que clasificamos en Secuenciales, de decisión o repetición:

- Las secuencias son generalmente instrucciones de entrada y salida
- Las condicionales, También llamadas de decisión nos permite dada una condición modificar el flujo de ejecución del programa

- Las de repetición: dado un bloque de código lo Podemos ejecutar varias veces dependiendo de una condición

Solo para ubicarnos en el universo de lenguajes de programación, Python se encuentra en el grupo de lenguajes creado en la década de los 80. Toma sus raíces en el lenguaje ABC pero También de otros como modula. Si, aunque no lo parezca Python es un lenguaje nuevo. La primera versión se presentó como la 0.9.0, luego más tarde agrego la posibilidad de trabajar con objetos y un llamativo control de excepciones. Hoy, actualmente, la última versión publicada es la 3.9.2

El IDE

Un *IDE* (Entorno de desarrollo integrado) es un entorno que facilita las tareas al programador. Es una herramienta que brinda un conjunto interesante de servicios que transforman el trabajo arduo del programador en una tarea un poco más razonable. Gracias al *IDE* el programador solo se preocupa en resolver problemas y no en como convertir una idea en código binario. En pocas palabras, mejora el potencial del programador permitiéndole concentrarse solo en el importante … programar que Brinda en tiempo real ayuda para el programador. El momento de elegir cual es el IDE de trabajo, muchas veces depende del lenguaje o de la plataforma donde trabajamos (Wiindows, Mac o linux). Pero ya existen IDES multiplataforma lo cual torna la decisión en algo más sencilloProgramas y módulos.

Forma de programar en Python.

Esto será una pequeña introducción, y aunque no lo crean es muy importante. Todos los lenguajes son iguales en términos de instrucciones o el set de instrucciones. Pero, cada lenguaje tiene su exigencia particular. Python es uno de esos lenguajes, ¿por qué? … porque a diferencia de lenguajes como C, las instrucciones compuestas no tienen ningún carácter. En Python, se representa con 4 espacios o un doble tab.

En consecuencia, vamos a tener que estar entrenados frente a esta característica que tiene el lenguaje. Pero, no nos vamos a preocupar por ello ahora. En el corto plazo, será muy simple para todos.

Depuración.

Son incontables los servicios que puede dar le IDE, estos servicios puede ser tan simples como grabar y recuperar el código, o tan complejos como administrar el numero numerosos archivos de un proyecto. Pero dentro de las ventajas de un ide Podemos contar con la posibilidad de Testear el programa (Debug) en vivo siguiendo el código línea por línea.

Variables

Python es un lenguaje de programación creado en los principios de los 90, con un nombre inspirado en un grupo de cómicos. Es un lenguaje con una sintaxis muy limpia que favorece la legibilidad.

Es un lenguaje interpretado o de Script, con tipado dinámico, multiplataforma y orientado a objetos. El código fuente de Python, se traduce a un lenguaje intermedio llamo bytecode generando archivos .pyc o pyo.

¿Qué es el tipo dinámico?

Las variables en Python no necesitan declaración de tipo, sino que el tipo se define en tiempo de ejecución. El tipo de valor de la variable puede cambiar si se le asigna un valor nuevo. Es fuertemente tipado, lo que significa que tenemos que convertir la variable si la queremos usar como otra cosa. Es decir, si la variable es de tipo texto, no puedo tratarla como número.

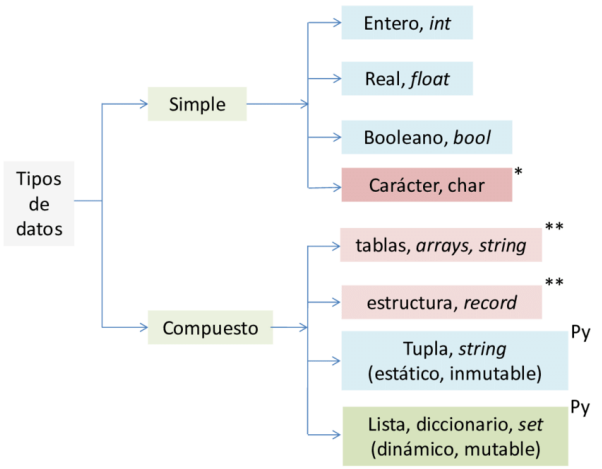
A veces nos peguntamos, como Podemos lograr que la computadora sume dos números sin saber cuáles son esos números. La manera es simple, usamos comodines. En Realidad, son auxiliares que representa temporalmente el dato, con lo cual, podemos hacer totos los cálculos asumiendo que ese auxiliar es el dato que vamos a usar.

Las variables, se guardan en posiciones de memoria. Y la cantidad de Byte que ocupan en la memoria, depende desde ya por el dato que se desea guardar. Cuando hablamos de dato, hablamos del tipo de dato. Mas adelante vamos a charlar sobre eso.

Algo que es muy importante recordar, es que una variable solo puede almacenar un dato a la vez. No parece importante ahora, pero te acordaras de esto seguramente más adelante.

Existen algunos **acuerdos** para nombrar variables, tenemos aquellos que son limitaciones por las reglas del mismo lenguaje y otros que son simples acuerdos entre programadores. Como se trata de un lenguaje de tipado dinámico, algunos proponen agregar al nombre de la variable el tipo de dato. Al principio parece una muy buena idea, pero es cuestión de gustos.

Un lenguaje de programación es dinámicamente **tipado** si una variable puede tomar valores de distinto tipo. La mayoría de los lenguajes de **tipado dinámico** son lenguajes interpretados, como **Python** o Ruby. Un lenguaje que no es dinámicamente **tipado** se dice que es de **tipado** estático, o estáticamente **tipado**.



Python, al igual que JavaScript, es un lenguaje de tipado dinámico. Hay dos formas de tipado:

- Estático
- Dinámico

En los lenguajes estáticos, cuando declaramos una variable es necesario especificar el tipo que tiene y ese tipo es inalterable:

String nombre = 'Héctor'; //

java En cambio en lenguajes como Python no es necesario establecer un tipo y además puede cambiar en cualquier momento, eso es porque viene determinado en tiempo de ejecución por el valor asignado a la variable:

nombre = 'Héctor’

Pasando la variable o literal a la función type() de Python, podremos consultar el tipo de una variable, eso sí, tendremos que pasarlo a un print() para ver el resultado.

Entrada de datos por teclado

Veamos como leer un dato por teclado, es mas amplio que el ejemplo que les doy... pero esto es solo un comienzo. Entrada de valores por teclado, ten en cuenta que puede arrojar errores según como tratamos la entrada de datos.

```
c= input("dime algo ")
```

Salida de datos por Pantalla

Si tenemos una entrada de datos, tenemos una salida. En este caso, la salida es por pantalla. Les dejo el mismo comentario, esto es sólo un inicio.

```
c= input("dime algo ")

print (c)
```

Comentarios

- De una sola línea

```
# esto es un comentario
```

- Multilínea

```
”””” esto es un comentario
De muchas líneas””””
```

Importante: los comentarios en la misma línea del código deben separarse con dos espacios en blanco. Luego del # debe ir un solo espacio en blanco

No hay mucho para decir, en algunas prácticas de programación es muy común tener más documentación interna que externa. En Python, tenemos dos formas diferentes de agregar comentarios que no serán tomados en cuenta por el intérprete del lenguaje.

Bibliografía

- Varó, A. M., Sevilla, P. G., & Luengo, I. G. (2014). Introducción a la programación con Python 3. Recuperado de http://dx.doi.org/10.6035/Sapientia93