

# LOS 10 MEJORES LINK DE PYTHON

NUMERO 1



LIBROS PARA  
INICIAR CON PYTHON

# MUNDO PYTHON



LIBRERIA  
3D PARA  
PYTHON

## Contenido

- AUTOR
- QUE ES PYTHO
- NOVEDADES ( LIBRERIA 3D PARA PYTHON)
- AUDIOCITY
- PYTHON PORTABLE
- LOS 10 MEJORES LINK DE PYTHON
- TKINTER
- LIBROS PARA INICIAR CON PYTHON
- CODIGO FUENTE
- INTRODUCCION A LA PROGRAMACION
- CONTACTO
- PUBLICIDAD



## AUTOR

Hola a todos

Bueno la verdad no soy muy bueno para escribir, cual es la idea de haber comenzado esta revista, reunir la comunidad mas grande de Programadores de Python de habla, Español.

Como empezo esta iniciativa pues al buscar documentacion de Python no encotramos mucha, pero si hay una gran cantidad de personas con conocimientos en Python muy grandes que pueden compartir a esta comunidad que inicia, si eres un programador y te consideras excelente lo invito a hacer sus aportes para esta comunidad que inicia, hay muchos que tal vez no tienen idea de programacion y mucho menos en Python la idea es que por medio de este medio poder difundir la enseñanza de este lenguaje "que por cierto no es muy dificil de aprender" por eso lo invito amigo lector a hacer su contribucion.

Con esta introduccion quiero invitarlo a participar en esta gran comunidad que inicia ya son muchos los que se han registrado en el foro, el foro es la manera en que invito a dar sus contribuciones tenemos una gran biblioteca y estamos en inicio de un foro dedicado a los codigos fuentes tambien preguntas frecuentes y mucho mas, otra manera en que lo invito a participar es enviando alguna noticia o articulo que considere importante para el crecimiento de esta revista puede enviar su aporte a este correo y lo estudiaremos y si es bueno lo publicaremos.

Este es el correo para cualquier inquietud, sugerencia o aporte que quiera brindar: [aprenderpython@gmail.com](mailto:aprenderpython@gmail.com)

Bueno como les decia yo invito tanto a novatos como expertos en la programacion de Python a contribuir a esta comunidad que inicia tal vez no somos muchos pero con sus aportes podemos hacer grande esta comunidad de difusion de Python, gracias a que es software libre seguimos sus filosofia de hacer libre esta revista.

Disculparan el diseño de la misma pero como ven estamos iniciando, un saludo a todos y

Bienvenidos.

Daniel Bermudez.





## QUE ES PYTHON

Se compara habitualmente con TCL, Perl, Scheme, Java y Ruby. En la actualidad Python se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation. La última versión estable del lenguaje es actualmente la 2.5.2 (22 de febrero de 2008) (Se anunció la llegada de la versión 3.0 para agosto de 2008).

Python es considerado como la "oposición leal" a Perl, lenguaje con el cual mantiene una rivalidad amistosa. Los usuarios de Python consideran a éste mucho más limpio y elegante para programar.

Python permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender Python). También hay módulos incluidos que proporcionan E/S de ficheros, llamadas al sistema, sockets y hasta interfaces a GUI (interfaz gráfica con el usuario) como Tk, GTK, Qt entre otros...

Python es un lenguaje de programación creado por Guido van Rossum en el año 1990.

Hola mundo en: **Python**

```
print "Hola
mundo"
```

Python es un lenguaje interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa. También es una calculadora muy útil.

El nombre del lenguaje proviene de la afición de su creador original, Guido van Rossum, por los humoristas británicos Monty Python.<sup>2</sup> El principal objetivo que persigue este lenguaje es la facilidad, tanto de lectura, como de diseño.

## **LIBRERIA 3D PARA PYTHON**

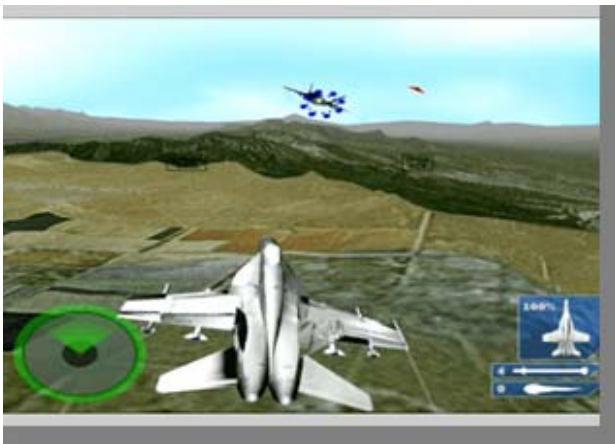


### **Qué es Panda3D**

Panda3D es un sistema 3D enfocado a la creación de videojuegos en PC.

Es una biblioteca de funciones escritas en C++ que sirven para simplificar grandemente la creación de videojuegos 3D a través de un lenguaje de programación. Se pueden usar estas funciones directamente en C++; pero si no te gusta C++, puedes programar tu juego usando Python, el cual es mucho más limpio y cómodo que C++ a la hora de programar la lógica del juego (los movimientos de los personajes, respuesta a los controles, máquinas de estados finitos, etc.).

Lo mejor de todo es que es libre y de código abierto. No tienes que pagar por usarlo ni nada de eso (ver en la licencia de uso lo que puedes y no puedes hacer con Panda3D).



## LIBRERIA 3D PARA PYTHON

### Qué no es Panda3D

Panda3D no es un programa para hacer videojuegos al estilo RPG Maker. Si quieres usarlo tendrás que aprender a programar.

Si ya sabes programar verás que usar Panda3D a través de Python no es nada difícil



### ¿Por qué Panda3D?

¿Por qué Panda3D y no otros sistemas open source populares como OGRE, Irrlicht, Crystal Space, etc...?

Primero que nada, a diferencia de Irrlicht y OGRE, Panda3D no solo es una api para renderear escenas 3D en tiempo real, sino que es un sistema completo que trae el sonido, física (muy básica, pero dudo que alguno de nosotros necesite algo más potente que lo que viene), manejo de tiempo y demás ya integrado, por lo que tu labor se reduce a simplemente programar tu juego, no la infraestructura necesaria para hacerlo.

Así que aquí ya no hay necesidad de integrar el sistema de gráficos con un sistema de física, de entrada/salida, un lenguaje de scripting, actores y muchísima infraestructura más que se requiere para hacer un juego de una calidad aceptable en un tiempo razonable.

## **LIBRERIA 3D PARA PYTHON**



Ya se han hecho videojuegos con el sistema. De hecho, este sistema nace gracias a un MMORPG que creó Disney, el cual se llama Toontown.

Disney después libera el código fuente del engine que crearon para hacer este juego para que las universidades pudieran mejorarlo. El Carnegie Mellon Entertainment Technology Center lo tomó y lo mejoró mucho en cuanto su facilidad de uso porque está pensado para que sus estudiantes hagan juegos de forma rápida.

Los desarrolladores principales actualmente son Disney, a quien le interesa la potencia para hacer juegos comerciales, y el ETC, a quien le interesa la facilidad de uso para sus estudiantes.



El resultado no podría ser mejor: Es una de las pocas cosas que encontrarán para hacer 3D \*casi\* tan fácilmente como en el RPG Maker y con la suficiente potencia como para que no sea herramienta para la creación de Videojuegos en union con el lenguaje Python.



## LIBRERIA 3D PARA PYTHON

La instalación es bien fácil. Es tan simple como bajar el archivo de instalación (Winblows) y darle doble click.

No necesitan bajar y configurar Python, pues eso lo hace el instalador por sí mismo. Python es más limpio que C++, por lo que brincar de RGSS a Python es solo cuestión de aprender las facilidades que te brinda Python. Otra cosa que lo hace distinto es la rápida curva de aprendizaje. Una vez que conocen los conceptos básicos de la programación 3D (eso se aprende aparte), aprender a usar Panda3D toma MUY poco tiempo, puedo decirles que cosa más fácil de usar no hay.



Una cosa muy importante que lo hace diferente de otros engines 3D, es que ya se han hecho videojuegos con esto. Así que si necesitan referencia sobre cómo hacer uno, simplemente bajan el código fuente del juego Airblade (escrito en Python) y lo revisan.

**Fuente** [Foros de RPGMAKERXP.COM](http://Foros.de.RPGMAKERXP.COM)

## AUDIOCITY



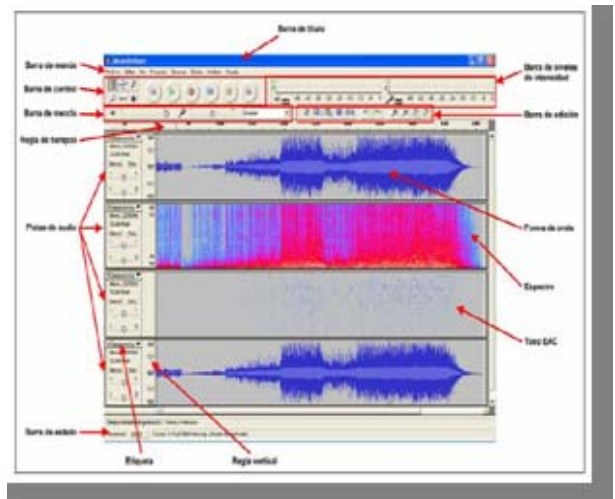
Audacity es un programa multiplataforma de grabación y edición de sonidos fácil de usar, de libre uso y de código abierto distribuido bajo licencia GPL. Debido a su calidad ha sido introducido en numerosas distribuciones GNU/Linux al ser uno de los programas libres de edición de sonido más fiable y avanzado que existe actualmente.

Existe una versión portátil de Audacity que puede ser transportada y usada directamente desde una memoria USB sin necesidad de instalarse en el computador.

### Ventajas

Las ventajas que tiene son:

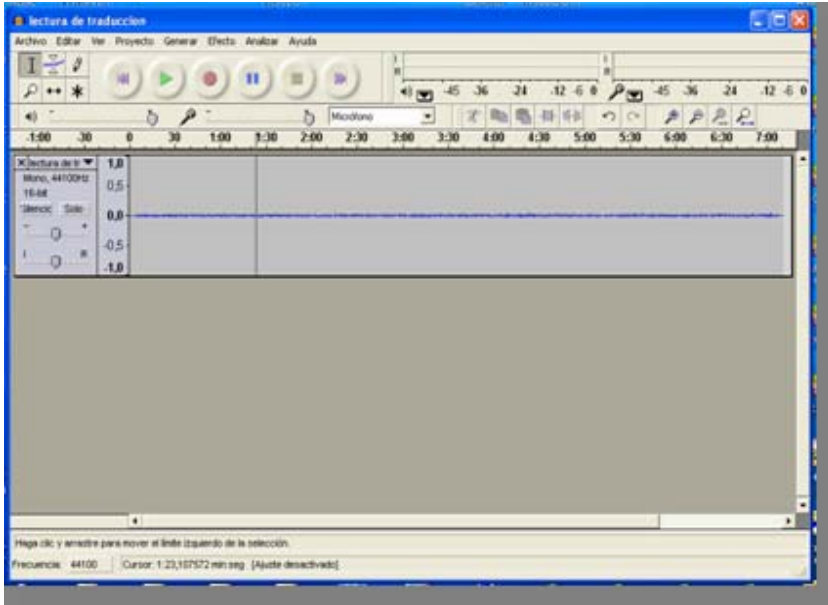
- ✓ Es software libre, en continua evolución
- ✓ Está disponible en castellano
- ✓ Es fácil de usar
- ✓ Es multiplataforma
- ✓ Se puede descargar desde la web
- ✓ Se puede conseguir a un bajo/nulo costo
- ✓ Es extensible / programable mediante plugins



\*Como hacer Audio Podcast con Audacity

[AQUI](#)

## AUDICITY



\*Manual de Audacity

[AQUI](#)

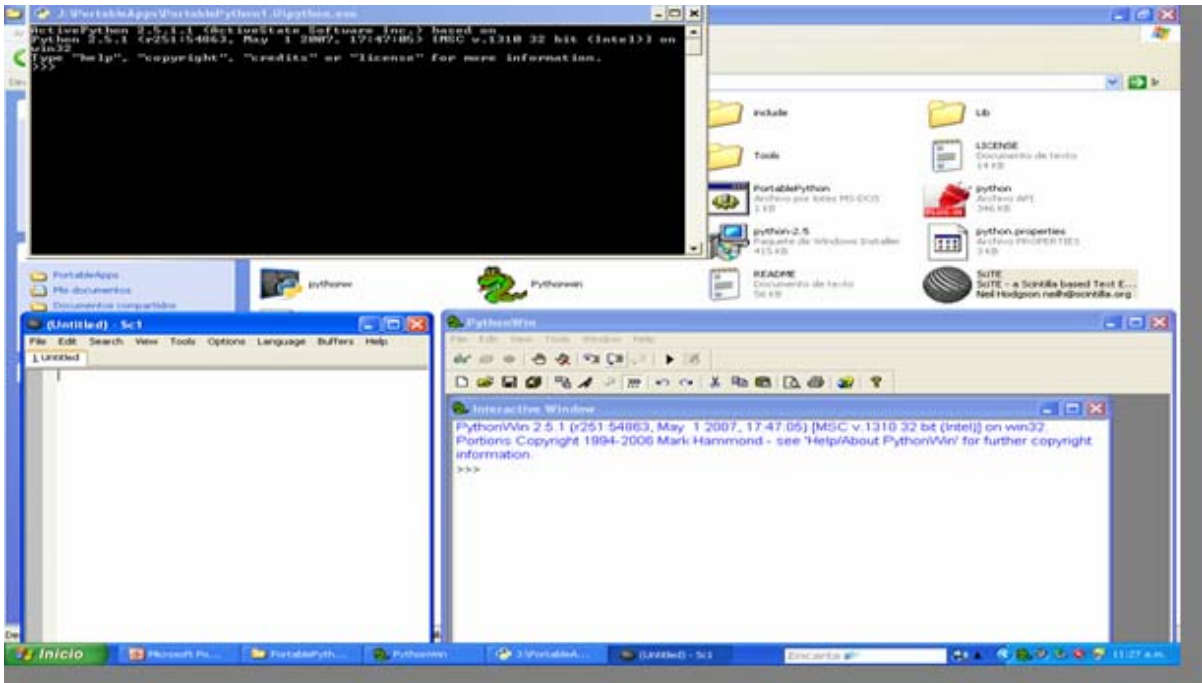
\*Manual de Edición de Sonido con Software Libre

[AQUI](#)

### Posibilidades de uso

- Grabar sonido en directo.
- Convertir cassettes y grabaciones analógicas en grabaciones digitales.
- Convertir entre formatos de audio tipo MP3, Ogg Vorbis y WAV
- Editar archivos de audio tipo Ogg Vorbis, MP3, WAV, AIFF, AU y LOF.
- Importar archivos de formato MIDI y RAW.
- Permite, mediante su interfaz gráfica cortar, copiar, pegar y mezclar sonidos.
- Puede trabajar con múltiples pistas.
- Eliminar ruidos, normalizar, ecualizar, amplificar, cambiar velocidad y modificar tonos.
- Es posible programar cualquier efecto musical o sonoro, mediante el panel Nyquist, usando Xlisp.

## Python Portable



Muchas veces necesitamos un programa y no dispones del programa en el PC donde nos encontramos.

Para eso estan los Portables que son: Programas que se ejecutan sin estar instalados en un Computador en especifico solo tienes que tener el programa en una memoria USB o dispositivo USB con el programa portable, simplemente das doble clic y el programa ya esta funcionando.

Python tambien tiene una distribucion con estas mismas caracteristicas Portable para ello simplemente hay que descargarlo desde esta disponible en esta pagina, como ven en la imagen Python dispone de un entorno como DOS y otro mas grafico todo esto en un solo portable, dispones de muchas mas caracteristicas en esta version portable.

**Python Portable**

**AQUI**

## 10 MEJORES LINK DE PYTHON



### Active Python

Este enlace es una distribución de Python exclusivo para Windows que contiene muchas más utilidades que el Python de la Página Original, dentro de su contenido, extensiones zlib y bzip2, la biblioteca bsddb, los widgets TiX GUI para Tkinter, el IDE PythonWin, documentación adicional, etc. Enlace [AQUI](#)



### Tkinter

En este enlace van a encontrar la distribución de la página oficial, esta es la distribución original con la cual viene Python. Su enlace [AQUI](#)



### Iron Python

IronPython es una implementación del intérprete Python (cPython) escrita totalmente en C#. El proyecto trata de seguir al pie de la letra el lenguaje Python, como implementación de Python que es. Esto hace que cualquier programa escrito en Python pueda ser interpretado con IronPython, enlace [AQUI](#)



### Jython

Es un lenguaje de programación de alto nivel, dinámico y orientado a objetos basado en Python e implementado en Java (100%), su antecesor fue JPython, Jython al igual que Python es un proyecto de código libre. Enlace [AQUI](#)



### Python.NET

Una integración entre el Python y el CLR, aunque no se podían compilar aplicaciones ni acceder a las clases de Python desde otros lenguajes de .NET., su enlace [AQUI](#)

## 10 MEJORES LINK DE PYTHON



### Python Win

Esta portado especialmente para windows (Win32-API) y te permite usar muchas funciones de estas integradas en el lenguaje de programación, enlace [AQUI](#)



### WXpython

wxPython es un juego de herramientas GUI para Python, que permiten crear aplicaciones con interfaces gráficas. Funciona como un módulo de extensión que encapsula la biblioteca gráfica wxWidgets. Enlace [AQUI](#)



### Boa Constructor

Boa Constructor es un IDE de Python y un constructor de GUI para wxPython. Enlace [AQUI](#)



### Python G

**PythonG** es un sencillo entorno de programación/ejecución para una versión extendida del lenguaje [Python](#). Dicha extensión consiste en incorporar al lenguaje una pequeña capacidad gráfica mediante un conjunto de funciones predefinidas. Enlace [AQUI](#)



### MacPython

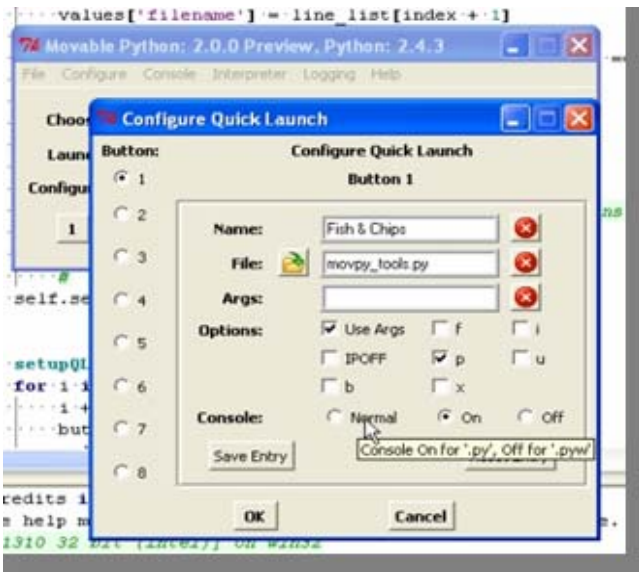
Al igual que su nombre es una distribución de Python exclusiva para equipos Macintosh, lo puede conseguir [AQUI](#)

## Tkinter

### Tkinter

Se puede decir que es el estándar en Python. Se distribuye junto con el propio intérprete de Python, es multiplataforma y está muy bien documentado. Un excelente lugar donde conocer Tkinter es este o bien dentro de la propia web de Python, <http://www.python.org/>

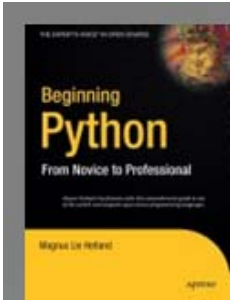
En mi opinión su mayor ventaja es que se distribuye junto con el intérprete. Es muy sencillo de aprender, si ya sabes Python, escribir en Tkinter una aplicación con una mínima funcionalidad no te llevará más de un par de horas (eso es lo que me llevó a mí programar esto sin saber nada de Tkinter y soy bastante lento).



Su instalación es muy fácil su uso también es muy fácil, no tiene mucha complejidad la verdad es una distribución buena pero la verdad yo preferiría otras distribuciones con entornos más agradables y con muchas más herramientas que permiten un mejor aprendizaje del lenguaje Python ya veremos en otra oportunidad una distribución que yo aconsejo para iniciar con Python.



## LIBROS PARA INICIAR CON PYTHON



### BEGINNING PYTHON FROM NOVICE TO PROFESIONAL

Bueno la verdad este libro es uno de los mejores que conozco para iniciar en Python es escrito por un profesor Universitario, ademas tiene muchos ejemplos y es en un lenguaje muy claro y facil de entender, dispones de 10 proyectos utiles, el unico inconveniente del libro es que es escrito en INGLES, Este enlace lo llevara al lugar donde encontrar sus características [AQUI](#)



### APRENDER A PENSAR COMO PROGRAMADOR CON PYTHON

Este libro es uno de los mas claros y paso a paso que conozco sobre programacion es practicamente un libro que no puede faltar dentro de la coleccion de libros para aprender a programar con Python, y lo mejor de este libro es que es escrito en ESPAÑOL y gratis. Este enlace lo llevara al lugar donde lo puede [descarga](#)



### BEGINNING PYTHON

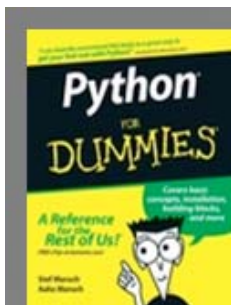
Este libro es uno de los mas recomendados para programar en Python es escrito por un grupo de programadores expertos en Python, tambien es un buen libro para iniciar en el mundo de la programacion con Python, este libro esta escrito en INGLES, en este enlace encontraran una referencia con respecto al libro [AQUI](#)

## LIBROS PARA INICIAR CON PYTHON



### INTRODUCCION A LA PROGRAMACION CON PYTHON

Bueno para empezar el libro es uno de mis recomendados, porque es un libro escrito por un profesor Universitario y además tiene una gran cantidad de ejemplos que dan claridad al aprendizaje de Python y lo mejor es en ESPAÑOL y gratis, puede descargarlo desde [AQUI](#)



### PYTHON FOR DUMMIES

Este libro es escrito al estilo de los libros para Dummies un lenguaje tan sencillo y fácil de entender que cualquier persona puede aprender a programar con este libro, el único problema es que viene en idioma INGLÉS pueden ver una referencia completa en este [Enlace](#)



### PYTHON PARA TODOS

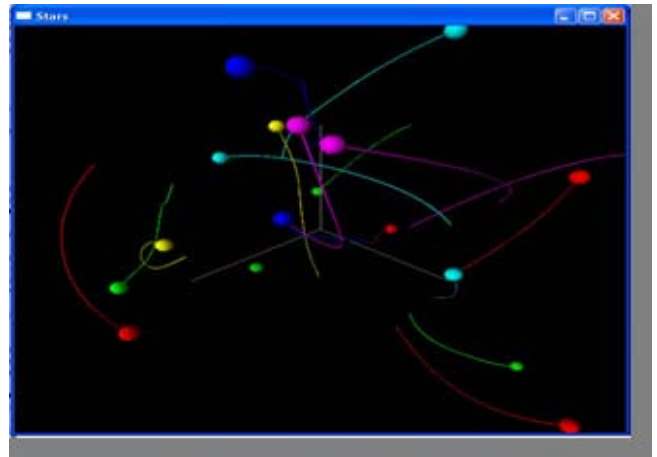
Este libro es uno de los más recientes libros que ha salido en idioma ESPAÑOL la verdad es un gran libro de programación en Python yo lo recomiendo al igual que los anteriores para iniciar con el mundo de la programación bajo el lenguaje Python. Tiene grandes ventajas uno el idioma dos es de lo más reciente y tres es gratis, puedes conseguirlo desde este [Enlace](#)

## CODIGO

```

from visual import *
from time import clock
from random import random
# Stars interacting gravitationally
# Program uses Numeric Python arrays for high speed computations
win=600
Nstars = 20 # change this to have more or fewer stars
G = 6.7e-11 # Universal gravitational constant
# Typical values
Msun = 2E30
Rsun = 2E9
Rtrail = 2e8
L = 4e10
vsun = 0.8*sqrt(G*Msun/Rsun)
print ""
Right button drag to rotate view.
Left button drag up or down to move in or out.
""
scene = display(title="Stars", width=win, height=win,
                range=2*L, forward=(-1,-1,-1))
xaxis = curve(pos=[(0,0,0), (L,0,0)], color=(0.5,0.5,0.5))
yaxis = curve(pos=[(0,0,0), (0,L,0)], color=(0.5,0.5,0.5))
zaxis = curve(pos=[(0,0,0), (0,0,L)], color=(0.5,0.5,0.5))

```

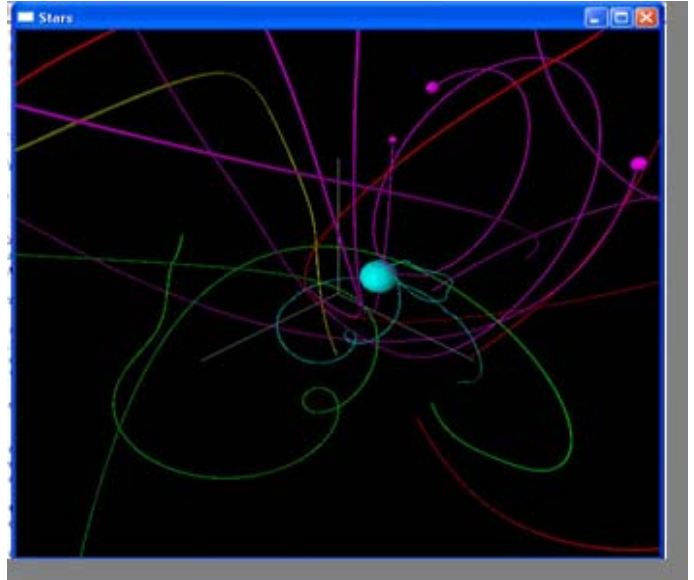


## CODIGO

```
fStars = []
colors = [color.red, color.green, color.blue,
          color.yellow, color.cyan, color.magenta]

poslist = []
plist = []
mlist = []
rlist = []

for i in range(Nstars):
    x = -L+2*L*random()
    y = -L+2*L*random()
    z = -L+2*L*random()
    r = Rsun/2+Rsun*random()
    Stars = Stars+[sphere(pos=(x,y,z), radius=r, color=colors[i % 6])]
    Stars[-1].trail = curve(pos=[Stars[-1].pos], color=colors[i % 6], radius=Rtrail)
    mass = Msun*r**3/Rsun**3
    px = mass*(-vsun+2*vsun*random())
    py = mass*(-vsun+2*vsun*random())
    pz = mass*(-vsun+2*vsun*random())
    poslist.append((x,y,z))
    plist.append((px,py,pz))
    mlist.append(mass)
    rlist.append(r)
```

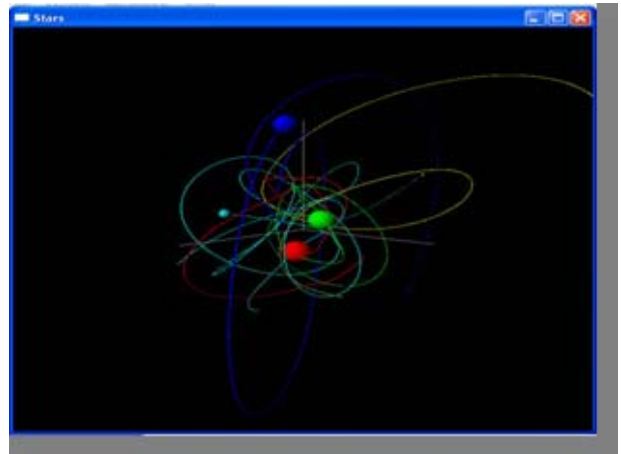


## CODIGO

```

pos = array(poslist)
p = array(plist)
m = array(mlist)
m.shape = (Nstars,1) # Numeric Python: (1 by Nstars) vs. (Nstars by 1)
radius = array(rlist)
vcm = sum(p)/sum(m) # velocity of center of mass
p = p-m*vcm # make total initial momentum equal zero
t = 0.0
dt = 1000.0
Nsteps = 0
pos = pos+(p/m)*(dt/2.) # initial half-step
time = clock()
Nhits = 0
while 1:
    # Compute all forces on all stars
    r = pos-pos[:,NewAxis] # all pairs of star-to-star vectors
    for n in range(Nstars):
        r[n,n] = 1e6 # otherwise the self-forces are infinite
    rmag = sqrt(add.reduce(r*r,-1)) # star-to-star scalar distances
    hit = less_equal(rmag,radius+radius[:,NewAxis])-identity(Nstars)
    hitlist = sort(nonzero(hit.flat)) # 1,2 encoded as 1*Nstars+2
    F = G*m*m[:,NewAxis]*r/rmag[:,NewAxis]**3 # all force pairs
    for n in range(Nstars):

```



## CODIGO

```

F[n,n] = 0 # no self-forces
p = p+sum(F,1)*dt
# Having updated all momenta, now update all positions
pos = pos+(p/m)*dt
# Update positions of display objects; add trail
for i in range(Nstars):
    Stars[i].pos = pos[i]
    if Nsteps % 10 == 0:
        Stars[i].trail.append(pos=pos[i])
# If any collisions took place, merge those stars
for ij in hitlist:
    i, j = divmod(ij,Nstars) # decode star pair
    if not Stars[i].visible: continue
    if not Stars[j].visible: continue
    # m[i] is a one-element list, e.g. [6e30]
    # m[i,0] is an ordinary number, e.g. 6e30
    newpos = (pos[i]*m[i,0]+pos[j]*m[j,0])/(m[i,0]+m[j,0])
    newmass = m[i,0]+m[j,0]
    newp = p[i]+p[j]
    newradius = Rsun*((newmass/Msun)**(1./3.))
    iset, jset = i, j
    if radius[j] > radius[i]:
        iset, jset = j, i
    
```



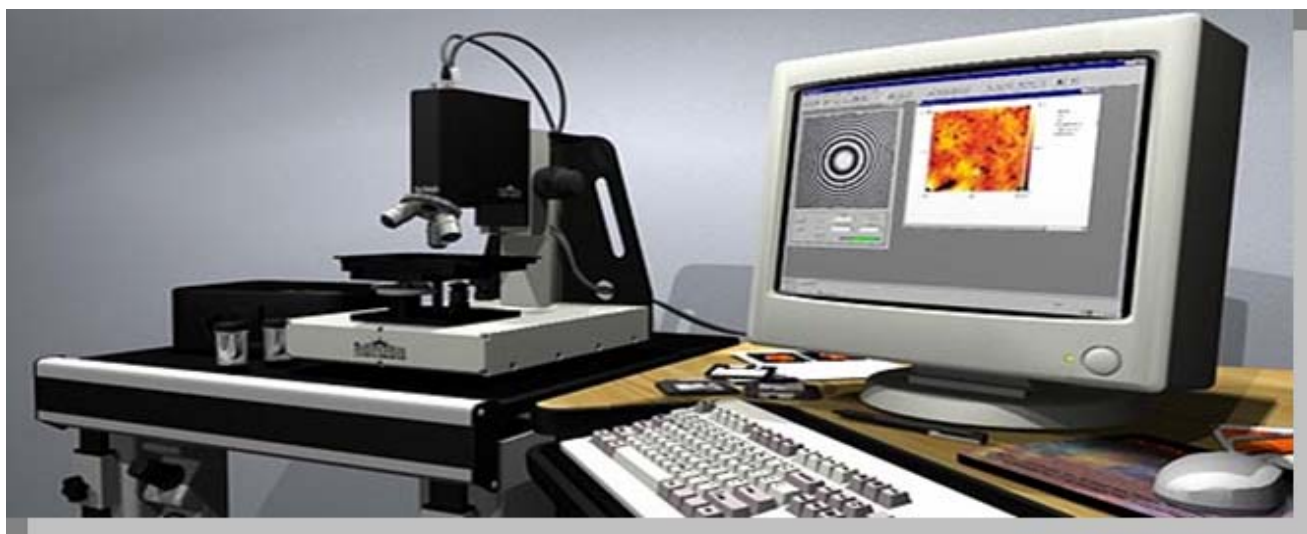
## CODIGO

```
Stars[iset].radius = newradius
    m[iset,0] = newmass
    pos[iset] = newpos
    p[iset] = newp
    Stars[jset].trail.visible = 0
    Stars[jset].visible = 0
    p[jset] = vector(0,0,0)
    m[jset,0] = Msun*1E-30 # give it a tiny mass
    Nhits = Nhits+1
    pos[jset] = (10.*L*Nhits, 0, 0) # put it far away

if Nsteps == 100:
    print '%3.1f seconds for %d steps with %d stars' % (clock()-time, Nsteps, Nstars)
Nsteps = Nsteps+1
t = t+dt
rate(100)
```



## INTRODUCCION A LA PROGRAMACION



Mucha gente piensa que estudiar metodología de la programación es una cosa ardua y muy aburrida. Intentare hacer que esto no sea así y que todo lo que aprendas a partir de este momento te sea de mucha utilidad para la creación de esos programas que tienes en mente.

Si todavía no ves claro el porqué de la metodología mira el siguiente ejemplo, después ya me dirás que opinas.

Imaginate que deseas realizar un programa sobre el calculo de horas a pagar de un trabajador promedio, y solo llegas a sentarte frente a tu computadora sin haber planeado como enfrentar el programa, llevas horas y horas ingresando codigo y probando los resultados. Luego de x horas te desesperas y aún no has llegado a resolver lo del pago de horas adicionales trabajadas !.

Aqui ves un ejemplo de lo que puede suceder si solamente llegas a escribir codigo y codigo sin haber planificado antes lo que harías en el programa.

## INTRODUCCION A LA PROGRAMACION

### La Resolución de Problemas utilizando la Computadora

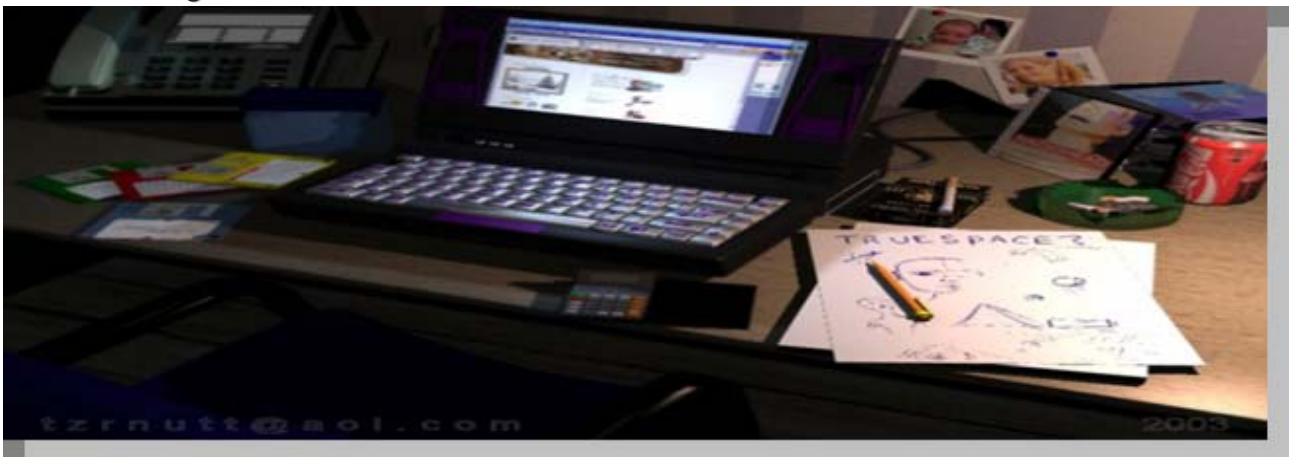
Aunque el proceso de diseñar programas es un proceso creativo, se pueden considerar una serie de fases o pasos comunes, que generalmente deben seguir todos los programadores.

La resolución de problemas con computadoras se pueden dividir en tres fases:

- Análisis del problema
- Diseño del algoritmo
- Resolución del algoritmo en la computadora

El análisis y el diseño del algoritmo requiere la descripción del problema en subproblemas a base de refinamientos sucesivos y una herramienta de programación:

- Diagrama de flujo
- Diagrama N-S
- Pseudocódigo



## INTRODUCCION A LA PROGRAMACION

Durante la tercera etapa se implementa este algoritmo en un código escrito en un lenguaje de programación, reflejando las ideas obtenidas en las fases de análisis y diseño. Antes de conocer las tareas a realizar en cada fase, definiremos el concepto y significado de la palabra algoritmo.

¿Qué es Algoritmo?

Se deriva de la traducción al latín de la palabra árabe Alkhowarismi, nombre de un matemático y astrónomo árabe que escribió un tratado sobre manipulación de números y ecuaciones en el siglo IX. Un algoritmo es un método para resolver un problema mediante una serie de pasos precisos, definidos y finitos.

Características del Algoritmo

- preciso, tiene que indicar el orden de realización en cada paso.
- definido, es decir, si el algoritmo se prueba dos veces, en estas dos pruebas, se debe obtener el mismo resultado.
- finito, es decir, que el algoritmo tiene que tener un número determinado de pasos.
- Debe producir un resultado en un tiempo finito.

Ejemplos de algoritmos son: Ver una película

1. Buscar el videocasette de la película
2. Si el televisor y la video se encuentran apagados, encenderlos
3. Sacar el video del estuche
4. Introducirlo en la videocaset era
5. Tomar el control del televisor y la video
6. Dirigirme a el sofa
7. Ponerme comodo
8. Disfrutar la película



## INTRODUCCION A LA PROGRAMACION

Fíjate como he descrito en estos pasos el algoritmo para poder ver una película en la video, este pequeño algoritmo cumple con los requisitos descritos arriba, ya que cada paso precisa un orden y tiene un orden de pasos finitos. En este algoritmo aparece la palabra **SI** remarcada en mayúsculas, el uso de esta palabra la veremos mas adelante, cuando discutamos sobre el control del flujo del programa o estructuras de control.

Los algoritmos se pueden expresar por fórmulas, diagramas de flujo, y pseudocódigos conocidos como herramientas de programación. Está última representación es la mas utilizada por su sencillez y parecido a el lenguaje humano.

Como ejercicio te recomendaría que escribieras algunos algoritmos de sucesos en tu vida cotidiana, como por ejemplo: encender el auto, ir al cine, etc..

### Fases para la Resolución de Problemas

En esta sección describire brevemente las fases o pasos a seguir para la resolución de problemas con ayuda de la computadora.

#### Análisis del Problema

Esta fase requiere una clara definición donde se contemple exactamente lo que debe hacer el programa y el resultado o solución deseada.

Dado que se busca una solución se precisan especificaciones de entrada y salida.

Para poder definir bien un problema es conveniente responder a las siguientes preguntas:

- ¿Qué entradas se requieren? (cantidad y tipo)
- ¿Cuál es la salida deseada? (cantidad y tipo)
- ¿Qué método produce la salida deseada?



## INTRODUCCION A LA PROGRAMACION

### Diseño del Algoritmo

En la fase de análisis en el proceso de programación se determina que hace el programa. En la fase de diseño se determina como hace el programa la tarea solicitada.

Los métodos utilizados para el proceso del diseño se basan en el conocido divide y vencerás. Es decir la resolución de un problema complejo se realiza diviendo el problema en subproblemas y a continuación dividir estos subproblemas en otros de nivel mas bajo, hasta que sea implementada una solución en la computadora. Este método se conoce técnicamente como diseño descendente (top-down) o modular.

Cada programa bien diseñado consta de un programa principal (el módulo de nivel mas alto) que llama a subprogramas (módulos) de nivel mas bajo, que a su vez pueden llamar a otros subprogramas.

Los módulos pueden ser planeados, codificados, comprobados y depurados independientemente y a continuación combinarlos entre sí. Este proceso implica la ejecución de estos pasos hasta que el programa se ha terminado:

- Programar un módulo
- comprobar el módulo
- Si es necesario, depurar el módulo
- Combinar el módulo, con el resto de los otros módulos

El diseño del algoritmo es independiente del lenguaje de programación en el que se vaya a codificar posteriormente.

### Implementación del Algoritmo

Para implementar un algoritmo en la computadora, se debe ejecutar los siguientes pasos:

- Codificación
- Compilación y ejecución
- Verificación
- Depuración



## INTRODUCCION A LA PROGRAMACION

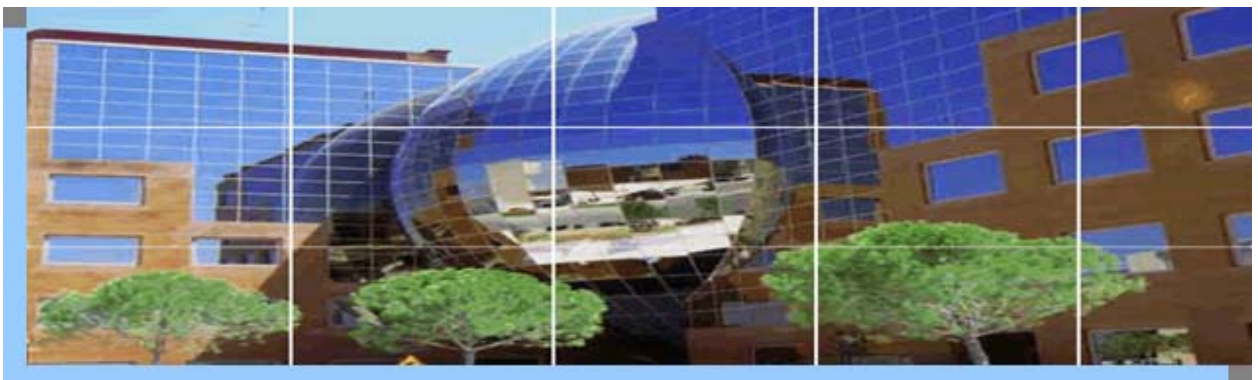
**Codificación:** Es la escritura en un lenguaje de programación de la representación de un algoritmo. Dado que el diseño del algoritmo es independiente del lenguaje de programación utilizado en su implementación, el código puede ser escrito con igual facilidad en un lenguaje o en otro.

**Compilación y ejecución:** Una vez que el algoritmo se ha convertido en un programa fuente, es preciso introducirlo en memoria mediante el teclado y almacenarlo posteriormente en un disco. Esta operación se realiza con un editor de texto, posteriormente el programa fuente se convierte en un archivo de programa que se guarda en un disco.

El programa fuente debe ser traducido a lenguaje máquina. Este proceso se realiza con el compilador y el sistema operativo que se encarga prácticamente de la compilación. Si al compilar el programa fuente se presentan errores (errores de compilación), es necesario volver a editar el programa, corregir los errores y compilar de nuevo. Esto se repite hasta que ya no se presenten más errores, obteniéndose el programa objeto, el cual todavía no es ejecutable directamente. Al ya no existir errores en el programa fuente se debe instruir al sistema operativo para que efectúe la fase de montaje o enlace, del programa fuente con las librerías del programa del compilador. Este proceso de montaje produce un programa ejecutable.

Cuando se ha creado un programa ejecutable este se puede ya ejecutar desde el sistema operativo con solo teclear su nombre.

Suponiendo que no existen errores durante la ejecución (errores en tiempo de ejecución), se obtendrá la salida de resultados correctos del programa.



## INTRODUCCION A LA PROGRAMACION

Verificación y depuración: Es el proceso de ejecución del programa con una amplia variedad de datos de entrada, llamados datos de test o prueba como son: valores normales de entrada, valores extremos de entrada que comprueben los límites del programa y valores de entrada que comprueben aspectos especiales del programa. Estos determinarán si el programa contiene errores o no.

Al ejecutar un programa se pueden producir tres tipos de errores:

- Errores de Compilación: Se producen normalmente por un uso incorrecto de las reglas del lenguaje de programación, suelen ser errores de sintaxis.
- Errores de Ejecución: Se producen por instrucciones que la computadora puede comprender pero no ejecutar. En estos casos se detiene la ejecución del programa y se imprime un mensaje de error. Ejemplo de esto puede ser una división por cero.
- Errores Lógicos: Se producen en la lógica del programa y la fuente del error suele ser el diseño del algoritmo, son mas difíciles de detectar puesto que el programa puede funcionar y no producir errores de compilación ni de ejecución pero regresará resultados incorrectos. En este caso se debe regresar a la fase de diseño, modificar el algoritmo, cambiar el programa fuente y compilar y depurar una vez mas.



Si quieres aprender mas acerca del tema de la programacion ingresa a la Biblioteca del Foro. [AQUI](#)

## INTRODUCCION A LA PROGRAMACION

**Documentación:** La importancia de la documentación debe ser destacada por su influencia en la etapa final, ya que programas pobremente documentados son difíciles de leer, mas difíciles de depurar y casi imposibles de mantener y modificar.

Puede ser interna y externa. La documentación interna es la contenida en líneas de comentarios. La documentación externa incluye análisis, diagramas de flujo y/o pseudocodigos, manuales de usuarios con instrucciones para ejecutar el programa y para interpretar los resultados.

La documentación es vital cuando se desea corregir posibles errores futuros o bien cambiar el programa. Estos cambios se denominan mantenimiento del programa.

Además es de buena costumbre para todo buen programador, dejar comentado su código, esto es para que el futuro programador pueda darl e mantenimiento fácilmente a el programa, o incluso, si es el mismo creador quien debe darle mantenimiento.

Si quieres aprender mas acerca del tema de la programacion ingresa a la Biblioteca del Foro.

[AQUI](#)



## CONTACTO

HOLA

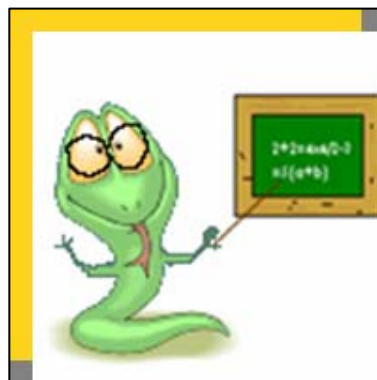
SI TIENES ALGUNA SUGERENCIA O ARTICULO QUE QUIERAS QUE SE PUBLIQUE EN LA PROXIMA EDICION, CON RESPECTO A LA REVISTA PUEDES ESCRIBIR AL CORREO QUE APACERECE ENSEGUIDA.

PARA LA PUBLICIDAD SI DESEAS ANUNCIARTE CON NOSOTROS SOLO DEBES ESCRIBIR AL CORREO, **¿PREGUNTARAS POR QUE ANUNCIAR CON NOSOTROS?**

- **ES UNA REVISTA LIBRE**
- **MUCHAS PERSONAS DESCARGAN LA REVISTA**
- **LOS COSTOS NO SON ALTOS**
- **DISPONEMOS DE PAGINA WEB**

AUTORES:

- **DANIEL BERMUDEZ**
- **JHON GARNICA**
- **DIEGO VARGAS**



- PAGINA WEB: [HTTP:WWW.APRENDERPYTHON.COM](http://www.aprenderpython.com)
- CORREO ELECTRONICO: [APRENDERPYTHON@GMAIL.COM](mailto:APRENDERPYTHON@GMAIL.COM)
- FORO: [WWW.APRENDERPYTHON.COM.SMF/](http://www.aprenderpython.com.smf/)
- PUBLICIDAD: [APRENDERPYTHON@GMAIL.COM](mailto:APRENDERPYTHON@GMAIL.COM)
- BLOG: [MUNDO PYTHON](http://MUNDOPYTHON)

## PUBLICIDAD

**CÓMO CONSEGUIR TU DIPLOMA AL FINALIZAR EL CURSO**

Podrás demostrar los conocimientos que has adquirido en cualquier país.

▶ Visita nuestra web [AQUI educahome.com](http://www.educahome.com)



**VIDEOTUTORIALES VMWARE** ahora en [www.biblio-web.org /aprendervmware](http://www.biblio-web.org/aprendervmware)

[AQUI](#)



**APRENDE PROGRAMACIÓN SIN ESFUERZO DESDE TU PC!**

▶ Visita nuestra web [AQUI educahome.com](http://www.educahome.com)



**APRENDE FOTOGRAFÍA DIGITAL SIN ESFUERZO DESDE TU PC!**

descubre los secretos de la imagen digital y sus técnicas más avanzadas

▶ Visita nuestra web [AQUI educahome.com](http://www.educahome.com)



**APRENDE 3D STUDIO** [AQUI](#)



[AQUI](#)



[AQUI](#)