Colección de actividades Aprender Conectados

Nivel Secundario

Programación

Tarjeta de amistad personalizada

Actividad N° 4

APRENDER CONECTADOS

Ministerio de Educación, Cultura, Ciencia y Tecnología
Presidencia de la Nación

Autoridades

Presidente de la Nación
Mauricio Macri

Jefe de Gabinete de Ministros
Marcos Peña

Ministro de Educación, Cultura, Ciencia y Tecnología
Alejandro Finocchiaro

Secretario de Gobierno de Cultura
Pablo Avelluto

Secretario de Gobierno de Ciencia, Tecnología e Innovación Productiva
Lino Barañao

Titular de la Unidad de Coordinación General del Ministerio de Educación, Cultura, Ciencia y Tecnología
Manuel Vidal

Secretaria de Innovación y Calidad Educativa
Mercedes Miguel

Subsecretario de Coordinación Administrativa
Javier Mezzamico

Directora Nacional de Innovación Educativa
María Florencia Ripani

ISBN en trámite

Este contenido fue producido por el Ministerio de Educación, Cultura, Ciencia y Tecnología de la Nación en el marco del Plan Aprender Conectados

Introducción

El Plan Aprender Conectados es la primera iniciativa en la historia de la política educativa nacional que se propone implementar un programa integral de alfabetización digital, con una clara definición sobre los contenidos indispensables para toda la Argentina.

En el marco de esta política pública, el Consejo Federal de Educación aprobó, en 2018, los Núcleos de Aprendizajes Prioritarios (NAP) de Educación Digital, Programación y Robótica (EDPR) para toda la educación obligatoria, es decir, desde la sala de 4 años hasta el fin de la secundaria. Abarcan un campo de saberes interconectados y articulados, orientados a promover el desarrollo de competencias y capacidades necesarias para que los estudiantes puedan integrarse plenamente en la cultura digital, tanto en la socialización, en la continuidad de los estudios y el ejercicio de la ciudadanía, como en el mundo del trabajo.

La incorporación de Aprender Conectados en la Educación Secundaria permite poner a disposición estudiantes y docentes, tecnología y contenidos digitales que generan nuevas oportunidades para reconocer y construir la realidad: abre una ventana al mundo, facilita la comunicación y la iniciación a la producción digital.

La sociedad está cambiando a un ritmo más acelerado que nuestro sistema educativo y la brecha entre las propuestas pedagógicas que presentan las escuelas y la vida de los estudiantes se amplía cada vez más. Garantizar el derecho a aprender en el siglo XXI implica que todos los estudiantes puedan desarrollar las capacidades necesarias para actuar, desenvolverse y participar como ciudadanos en esta sociedad cada vez más compleja, con plena autonomía y libertad.

En este marco, Aprender Conectados presenta actividades, proyectos y una amplia variedad de recursos educativos para orientar la alfabetización digital en la educación obligatoria en todo el país. La actividad que se presenta a continuación y el resto de los recursos del Plan, son un punto de partida sobre el cual cada docente podrá construir propuestas y desafíos que inviten a los estudiantes a disfrutar y construir la aventura del aprender.

María Florencia Ripani
Directora Nacional de Innovación Educativa

Objetivos generales

Núcleos de Aprendizajes Prioritarios

Educación Digital, Programación y Robótica – Educación secundaria	Ofrecer situaciones de aprendizaje que promuevan en los alumnas y alumnos: • La comprensión general del funcionamiento de los componentes de hardware y software, y la forma en que se comunican entre ellos y con otros sistemas, entendiendo los principios básicos de la digitalización de la información y su aplicación en la vida cotidiana. • El desarrollo de proyectos creativos que involucren la selección y la utilización de múltiples aplicaciones, en una variedad de dispositivos, para alcanzar desafíos propuestos, que incluyan la recopilación y el análisis de información. • La creación, la reutilización, la reelaboración y la edición de contenidos digitales en diferentes formatos, entendiendo las características y los modos de representación de lo digital. • La resolución de problemas a partir de su descomposición en partes pequeñas, aplicando diferentes estrategias, utilizando entornos de programación tanto textuales como icónicos, con distintos propósitos, incluyendo el control, la automatización y la simulación de sistemas físicos.

Objetivos de aprendizaje

Esta actividad permitirá introducir al lenguaje de programación Python y está orientada a desarrollar conocimientos iniciales vinculados con los siguientes objetivos de aprendizaje:

- Familiarizarse con la utilización de librerías.
- Profundizar los conceptos de entrada, transformación y salida de datos.
- Comenzar a trabajar con gráficos personalizados en la salida de datos.
- Realizar un proyecto que reciba datos por teclado, los transforme y los imprima en un mensaje personalizado dentro de una ventana con componentes gráficos.

Materiales y recursos

- Computadora.
- Python 2.x instalado.

Desafío

En esta actividad aprendemos a ingresar datos a través del teclado, realizar transformaciones numéricas sobre ellos, y a imprimir en pantalla el resultado, utilizando librerías externas para darle un formato visualmente atractivo.

< Inicio >

Disparador.
¿Tienes algún amigo que que conozcas hace muchos años?
¿Te gustaría sorprenderlo con una tarjeta digital programada por vos?

En esta actividad creamos un programa que nos permite interactuar de forma simple con el algoritmo. Al ser ejecutado, el programa pide el nombre de la persona a quien está dirigida la tarjeta y en qué año lo conociste. Luego, el programa dibuja en la pantalla una tarjeta personalizada con un mensaje que incluye la cantidad de años de amistad al dia de hoy.

Python, como todo lenguaje de programación, requiere de la práctica para dominarlo. Cada línea de código presente en esta actividad está allí para que la copien, prueben, modifiquen y vuelvan a probar con todas las variantes que se les ocurra. El error es un componente esencial del aprendizaje, reconocerlo y corregirlo forma parte de las habilidades indispensables en la programación en general. Mediante esta práctica, las alumnas/os y docentes refuerzan los conceptos presentados, desarrollan habilidades vinculadas con la programación (iteración, cooperación, detección de patrones), además de realizar su propia versión de la actividad, fomentando la creatividad.

En este documento las líneas de código son presentadas con el siguiente formato, para su fácil identificación y copiado:

```
# Soy un comentario en el código
print 'Soy una línea de código'
Soy una línea de código
```

Las líneas de color azul son la respuesta al código introducido en las líneas anteriores y se presentan como un ejemplo del resultado a obtener. No deben ser copiadas y ni ejecutadas en IDLE.

También se incluyen comentarios en gris, precedidos por el símbolo numeral. Estos comentarios son notas que dan claridad al código, pero que Python ignora y no son ejecutados.

El ejercicio se propone para ser escrito línea por línea, en IDLE. Es importante que todos los estudiantes estén frente a una computadora con IDLE, de Python, abierto al momento de comenzar la actividad y que, a medida que avanzan, ejecuten el programa para comprobar que su código esté bien y corregir errores, si los hubiere.

Al final de este documento se presentan todas las líneas del programa juntas, que pueden ser guardadas en un archivo desde IDLE para ser ejecutado como un programa.

< Desarrollo >

La creación de la tarjeta virtual se puede dividir en los siguientes pasos:

- Importación de librerías externas
- Recolección de datos a través del teclado
- Cálculo y transformación de los datos
- Configuración de la ventana
- Formato y escritura de texto
- Finalización de la tarjeta con detalles decorativos

El docente recuerda a los alumnos acerca de la importancia de utilizar los comentarios e invita colocar los códigos para que python reconozca todos los acentos y signos en los sistemas operativos Linux y Windows. También se sugiere ingresar los datos del autor y el nombre del programa.

```
# -*- coding: cp1252 -*-
# -*- coding: 850 -*-
# -*- coding: utf-8 -*-
# Creador: Pedro Perez
# Tarjeta Digital de Amistad
```

Recuerda utilizar comentarios de forma frecuente explicando tu código. ¡El programador que lo lea te lo agradecerá!

1) Importación de librerías externas

Uno de los beneficios de la programación es la posibilidad de reutilizar código. Una de las maneras de hacerlo es empaquetar una serie de funcionalidades en lo que se llama una biblioteca o librería de código. Estas librerías están pensadas para ser utilizadas por otros programas que necesiten de estas funcionalidades, ahorrando a su desarrollador horas de trabajo. Una de las ventajas de Python es su inmensa comunidad y esto hace que existan miles de librerías para resolver todo tipo de problemas.

Para esta actividad nos valdremos de las librerías “Time” (tiempo en inglés), que facilita varias funciones para trabajar con el tiempo, y “Turtle”, que sirve para crear gráficos, lo que nos permitirá darle un formato más bonito a nuestro mensaje.

La importación del código de las librerías es el primer paso a realizar en cualquier proyecto de python. Las siguientes líneas de código importan las librerías “Time” y “Turtle” a nuestro programa.

```
from time import *
from turtle import *
```

2) Recolección de datos a través del teclado

En las siguientes líneas, el programa nos muestra la consigna y nos presenta unas preguntas para guardar su resultado dentro de las variables “nombre” y “comienzo_amistad”. Utilizamos la función “print” y la función “raw_input” para recibir los datos a través del teclado.

```
print 'Elige una amiga o amigo a quien conozcas hace muchos años.'
nombre=raw_input('¿Cual es su nombre? ')
comienzo_amistad=raw_input('¿En qué año lo conociste? (escribe las 4
cifras)')
```

¿Se te ocurre cómo modificar los mensajes? ¿y cómo formular otras preguntas y guardar su resultado?

3) Cálculo y transformación de los datos

Ahora que tenemos los datos de nuestro agasajado dentro de las variables, vamos a realizar unas operaciones numéricas para deducir la cantidad de años que llevan de amistad. Primero necesitamos saber en qué año estamos, para luego restar el dato guardado, al año presente, y así obtener la cantidad de años. Para esto, utilizamos las funciones "strftime" y "gmtime" de la librería "time".

Los sistemas informáticos cuentan el tiempo como la cantidad de segundos transcurridos desde el primero de enero de 1970, momento también conocido como "epoch". La función "gmtime" transforma esta cantidad de segundos en un formato de tiempo leible por un humano (año, mes, día, hora, segundo, dia de la semana y día del mes). Al ser ejecutado sin parámetros, por defecto nos devuelve la hora actual. La librería "strftime" nos permite recortar solo la parte que nos interesa del resultado de "gmtime". En este caso, solo nos interesa el año.

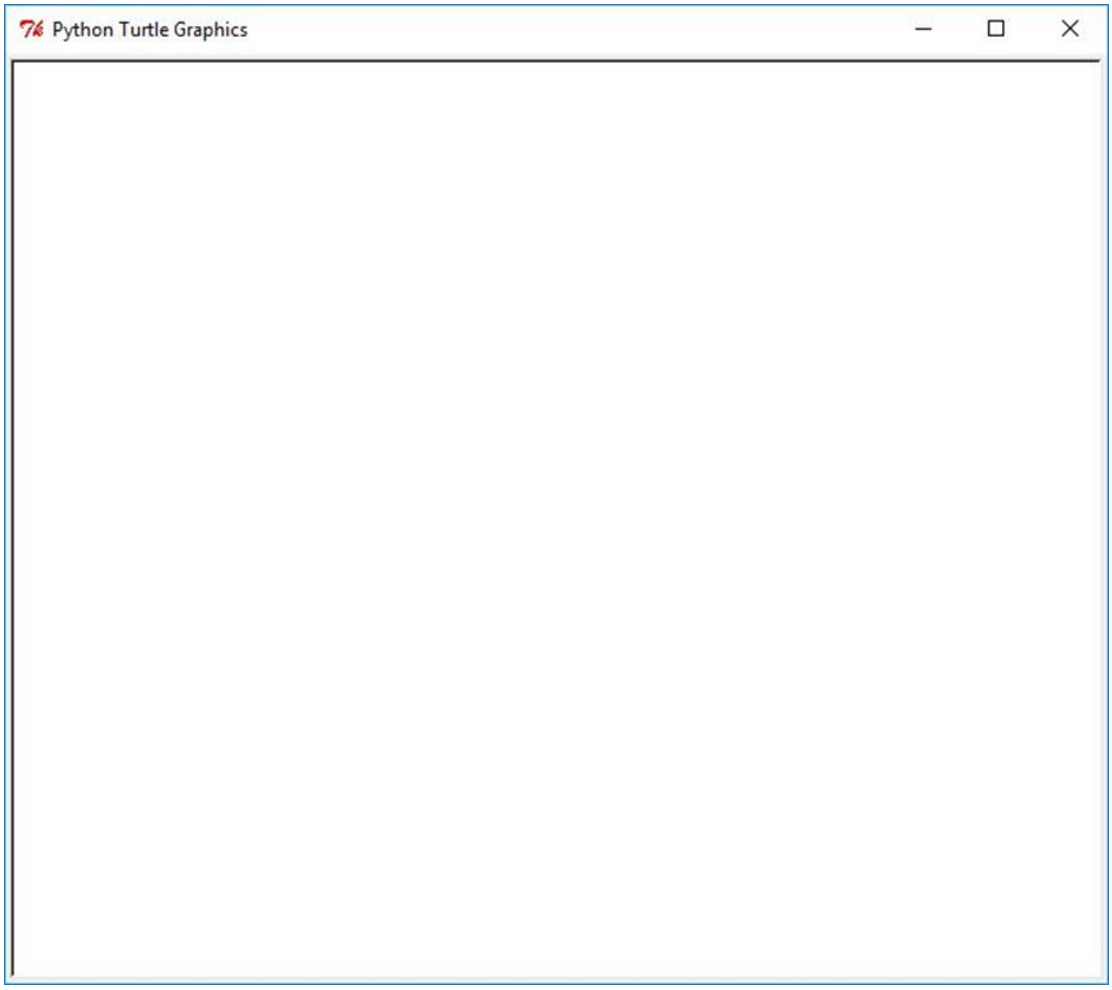
```
gmtime() # gmtime() nos devuelve el tiempo en un formato estructurado
time.struct_time(tm_year=2018, tm_mon=11, tm_mday=30, tm_hour=15,
tm_min=2, tm_sec=22, tm_wday=4, tm_yday=334, tm_isdst=0)
strftime('%Y', gmtime()) # strftime() recorta solo el dato que buscamos.
'2018'
ahora=strftime('%Y', gmtime()) # Lo guardamos en una variable.
cantidad_anios=int(ahora)-int(comienzo_amistad) # Cálculo.
```

¿Se te ocurre cómo obtener el mes y el dia actual? ¿Como usarias estos datos dentro del programa?

4) Configuración de la ventana

Ahora que tenemos calculada la cantidad de años de amistad, vamos a preparar la pantalla desde donde lo saludaremos. Para esto utilizamos algunas clases y funciones de la librería Turtle. Creamos un objeto "ventana" desde la clase "Screen". Una ventana blanca se crea en nuestro escritorio luego de ejecutar el comando. También definiremos el tamaño de la ventana en 700 píxeles por 700 píxeles para nuestro mensaje.

```
ventana=Screen()
ventana.setup(700,700)
```

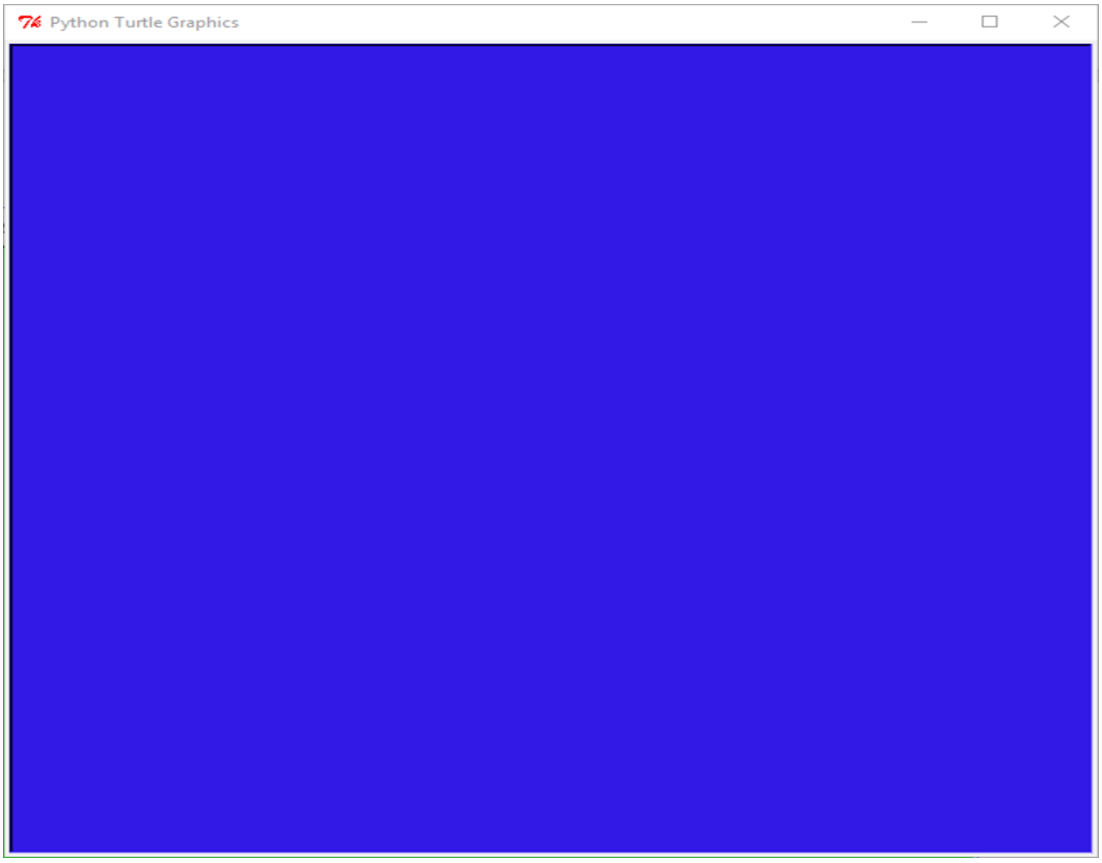


Resultado al ejecutar el código de ejemplo

Qué otros tamaños se te ocurren para tu tarjeta? ¿Qué pasa si tu mensaje es más grande que tu ventana?

Continuamos configurando la pantalla para acomodarla a nuestro gusto. Primero vamos a definir el rango de los colores primarios (rojo, verde y azul) entre 0 y 255. Esta es una forma muy popular de definir colores, y nos permite formar cualquier color de la paleta mediante la definición de la intensidad de cada uno de sus componentes primarios. En las siguientes líneas definiremos la intensidad para cada color dentro de una variable y usaremos la resultante para el fondo de nuestra tarjeta.

```
ventana.colormode(255) # Definimos el rango de intensidad hasta 255.
rojo=50 # cantidad de Rojo (entre 0 y 255)
verde=25 # cantidad de Verde (entre 0 y 255)
azul=230 # cantidad de azul (entre 0 y 255)
ventana.bgcolor(rojo,verde,azul)
```



Resultado al ejecutar el código de ejemplo

Experimenta con tus propios colores.

5) Formato y escritura de texto

Es momento de darle un formato al texto y escribir nuestro mensaje en la tarjeta.

Primero configuramos blanco como el color para el texto. Lo haremos utilizando colores primarios y la función “color” de Turtle. Ésta nos permite definir el color de nuestro texto.

Luego, definimos un estilo para la fuente, creando una lista de sus características. Para esto, en Python, se usan las listas, que son colecciones de datos ordenables y modificables, que se definen entre corchetes. En el ejemplo a continuación, definiremos la tipografía de la fuente, su tamaño y, finalmente, la haremos **negrita** para darle énfasis a nuestro saludo.

```
# Definimos blanco como el color del texto.
rojo=255
verde=255
azul=255
color(rojo,verde,azul)
# Definimos un estilo de fuente.
estilo=['Arial',30,'bold']
```

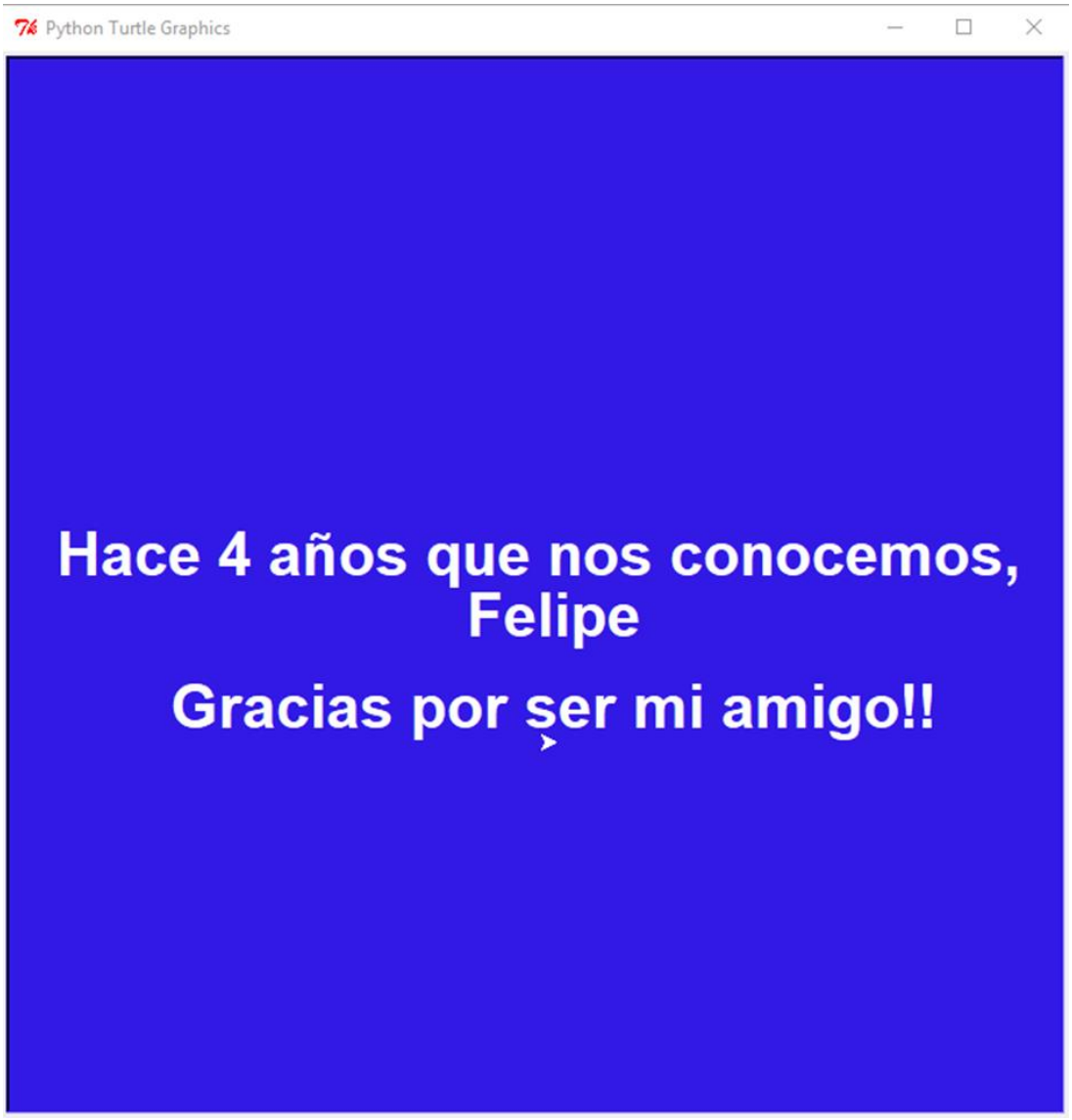
Para insertar nuestros datos dentro del mensaje, utilizaremos algunos trucos. Siempre que estamos programando, nos encontramos con la necesidad de unir o combinar cadenas de caracteres. A este proceso se le llama concatenar. En Python puedes concatenar líneas de texto entre comillas y el contenido de variables con la utilización del símbolo "+". Esta simpleza le da flexibilidad y agilidad al programador, a la hora de escribir su código.

A continuación escribimos tres líneas, una que dice "Hace hace XX años que nos conocemos," donde XX es reemplazado por la cantidad de años de amistad. Una segunda línea con su nombre y una tercera con la frase "Gracias por ser mi amigo!!". Para la primer frase utilizamos la variable "cantidad_anios" para reemplazar XX. Dado que esta variable es un valor numérico entero, debemos transformarlo a una cadena de caracteres primero utilizando la función "str". De esta manera podremos concatenarla al resto del texto utilizando el símbolo "+".

Escribimos el mensaje en la tarjeta utilizando la función "write" de Turtle, que recibe el mensaje que queremos imprimir, un estilo para la fuente, y la alineación del texto.

```
penup()
write("Hace " + str(cantidad_anios) + " años que nos
conocemos,",font=estilo,align='center') # Imprimimos la primer linea
goto (10,-40)
write(nombre,font=estilo,align='center') # La segunda es solo su nombre
goto(10,-100)
write("Gracias por ser mi amigo!!",font=estilo,align='center')
```

Las funciones "penup()" y "goto()" serán explicadas a continuación en la actividad. ¿Puedes imaginar que hacen? Prueba ejecutar el código sin ellas.



Resultado al ejecutar el código de ejemplo

6) Finalización de la tarjeta con detalles decorativos

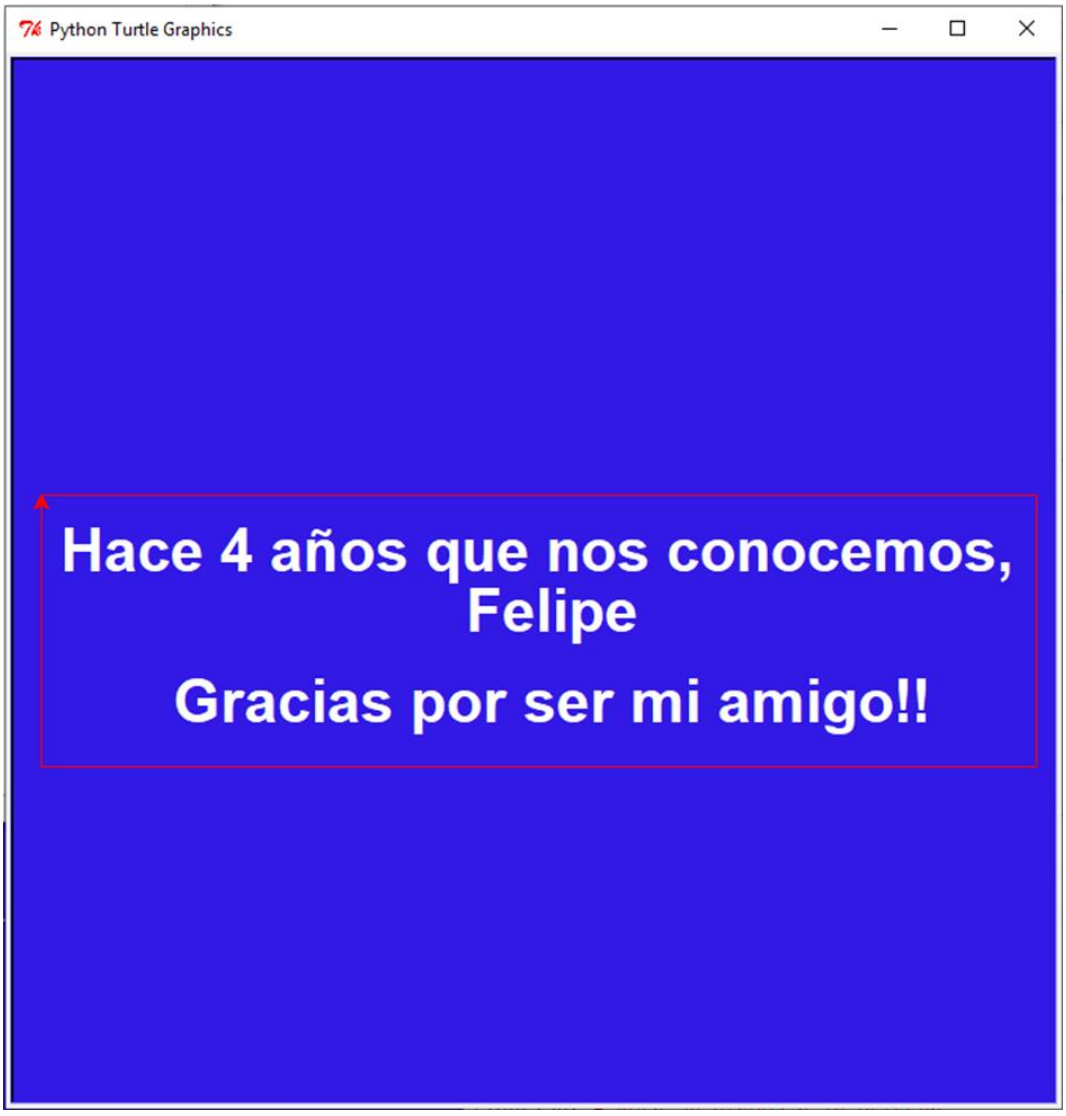
Imaginemos que tenemos una lapicera en la mano y debajo se encuentra una hoja. Si bajamos la mano y presionamos la lapicera contra la hoja, ésta la marca. Si movemos la mano, la lapicera dibujara el recorrido que hagamos. Si levantamos la mano, el dibujo se interrumpirá. Este mismo concepto utiliza Turtle para permitirnos dibujar utilizando la ventana como nuestra hoja. Con la función pendown() simulamos bajar la lapicera contra el papel. Con la función “penup()” simulamos levantar la lapicera de la hoja, con la función “goto(x,y)” le indicamos dónde posicionarse dentro de la ventana, indicando su posición horizontal (x) y vertical (y) como coordenadas dentro de un eje cartesiano.

Si al ejecutar la función “goto(x,y)” la lapicera se encuentra baja, se dibuja el recorrido desde la posición inicial hasta el destino. También podemos utilizar las funciones “forward()”, “backward()”, “right()” y “left()” para mover nuestra lapicera hacia adelante, atrás, girar a la derecha , o girar a la izquierda respectivamente.

Haremos uso de estas funciones para dibujar un rectángulo rojo enmarcando nuestro mensaje.

```
penup()
goto(-330,60) # Posiciona el cursor en la posición inicial
color('red') # Configura el color como rojo
pendown() # Baja la lapicera
forward(660) # Avanza 660 pasos
right(90) # Rota 90 grados a la derecha
forward(180) # Avanza 180 pasos
right(90) # Rota 90 grados a la derecha
forward(660) # Avanza 660 pasos
right(90) # Rota 90 grados a la derecha
forward(180) # Avanza 180 pasos completando el rectángulo
penup() # Levanta la lapicera
```

¿Notaste que utilizamos ‘red’ como parámetro para la función “color()”, en vez de sus componentes en colores primarios? Algunas funciones permiten más de un tipo de dato como parámetro.



Resultado al ejecutar el código de ejemplo

Finalmente, dibujaremos una carita feliz para decorar nuestra tarjeta. Para esto, utilizaremos la función “dot()” que dibuja en la ventana un punto del tamaño y color que le indicamos como parámetro. Utilizaremos esta herramienta de forma creativa para armar nuestra carita feliz.

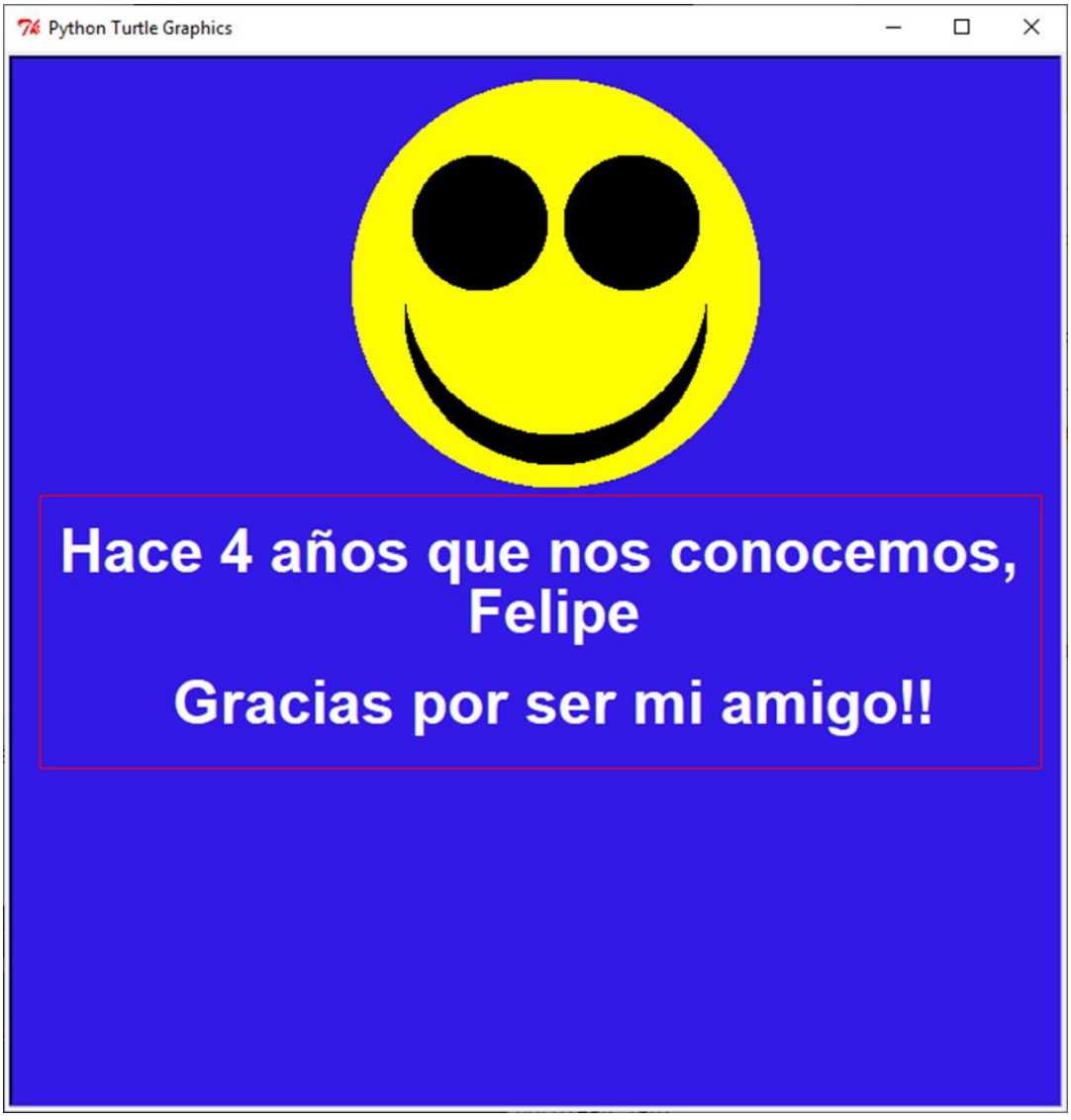
```
# Dibuja la cabeza
goto(10,200)
color('yellow')
dot(270) # crea un punto amarillo tamaño 270

# Dibuja la sonrisa
goto(10,180)
color('black')
dot(200) # crea un punto negro tamaño 200
goto(10,200)
color('yellow')
dot(200) # crea un punto amarillo tamaño 200

# Dibuja el ojo izquierdo
goto(-40,240)
color('black')
dot(90) # crea un punto negro tamaño 90

# Dibuja el ojo derecho
goto(60,240)
color('black')
dot(90) # crea un punto negro tamaño 90
```

¿Puedes explicar como se crea la carita feliz utilizando sólo puntos de colores? ¿Te animas a crear tu propio dibujo en la parte inferior de la tarjeta?



Resultado al ejecutar el código de ejemplo

¿Se animan a hacer cambios?

Código Completo

A continuación presentamos el código completo para ser copiado en un nuevo archivo de IDLE. Puedes guardarlo con el nombre tarjeta_digital.py

```
# -*- coding: cp1252 -*-
# -*- coding: 850 -*-
# -*- coding: utf-8 -*-
# Creador: Pedro Perez
# Tarjeta Digital de Amistad

# Incluimos las librerías que vamos a utilizar en nuestro programa
from time import *
from turtle import *

# Obtenemos los datos para personalizar la tarjeta
print 'Elige una amiga o amigo a quien conozcas hace muchos años.'
nombre=raw_input('¿Cual es su nombre? ')
comienzo_amistad=raw_input('¿En qué año lo conociste? (escribe las 4
cifras)')
ahora=strftime('%Y', gmtime()) # Guardamos el año actual en una variable
cantidad_anios=int(ahora)-int(comienzo_amistad) # Cálculo

# Inicializamos la ventana y su color de fondo
ventana=Screen()
ventana.setup(700,700)
ventana.colormode(255) # Definimos el rango de intensidad hasta 255
rojo=50 # cantidad de Rojo (entre 0 y 255)
verde=25 # cantidad de Verde (entre 0 y 255)
azul=230 # cantidad de azul (entre 0 y 255)
ventana.bgcolor(rojo,verde,azul)

# Definimos blanco como el color del texto para nuestro mensaje
rojo=255
verde=255
azul=255
color(rojo,verde,azul)
# Definimos un estilo de fuente
estilo=['Arial',30,'bold']

# Escribimos el mensaje en la tarjeta
penup()
write("Hace " + str(cantidad_anios) + " años que nos
conocemos,",font=estilo,align='center') # Imprimimos la primer línea
goto (10,-40)
write(nombre,font=estilo,align='center') # La segunda es solo su nombre
goto(10,-100)
write("Gracias por ser mi amigo!!",font=estilo,align='center')
```

```
# Dibujamos un rectángulo rojo alrededor de nuestro mensaje
penup()
goto(-330,60) # Posiciona el cursor en la posición inicial
color('red') # Configura el color como rojo
pendown() # Baja la lapicera
forward(660) # Avanza 660 pasos
right(90) # Rota 90 grados a la derecha
forward(180) # Avanza 100 pasos
right(90) # Rota 90 grados a la derecha
forward(660) # Avanza 340 pasos
right(90) # Rota 90 grados a la derecha
forward(180) # Avanza 100 pasos completando el rectángulo
penup() # Levanta la lapicera

# Pasos para dibujar la carita feliz

# Dibuja la cabeza
goto(10,200)
color('yellow')
dot(270) # crea un punto amarillo tamaño 270

# Dibuja la sonrisa
goto(10,180)
color('black')
dot(200) # crea un punto negro tamaño 200
goto(10,200)
color('yellow')
dot(200) # crea un punto amarillo tamaño 200

# Dibuja el ojo izquierdo
goto(-40,240)
color('black')
dot(90) # crea un punto negro tamaño 90

# Dibuja el ojo derecho
goto(60,240)
color('black')
dot(90) # crea un punto negro tamaño 90
```

Después de guardar el archivo, puedes ejecutarlo desde IDLE presionando la tecla “F5”.

< Cierre >

El docente pide a los estudiantes que se agrupen en parejas para probar sus proyectos. Si ven que algo del programa no funciona, trabajan juntos para encontrar el error y solucionarlo.
Luego, se sugiere hacer una puesta en común para intercambiar sobre lo que aprendieron, lo que más les gustó hacer y sobre los desafíos que se les presentaron.

Si se dispone de tiempo, o bien en un próximo encuentro, el docente puede proponer a los alumnos/as, crear nuevas tarjetas digitales en base a lo aprendido.

Evaluación:
El docente evalúa teniendo en cuenta que en esta actividad se incorporaron muchas instrucciones nuevas, lo cual requiere cierto tiempo de práctica y aprendizaje. La primera vez que se realice esta actividad, no se espera que todas las instrucciones estén bien aplicadas, sino que se puedan seguir los pasos a grandes rasgos, buscando que los estudiantes que muestran mayor facilidad, ayuden a los que más les cuesta este tipo de desafíos.
A continuación, se presentan preguntas orientadoras:

- ¿Lograron recolectar los datos a través del teclado?
- ¿Pudieron escribir los mensajes en pantalla?
- ¿Lograron dibujar el marco y la carita feliz?
- ¿Encontraron errores? ¿Pudieron solucionarlos?

El proceso de evaluación de la evolución del aprendizaje podrá continuar con la actividad propuesta a continuación.

Para seguir aprendiendo

Sugerimos al docente plantear una actividad en grupo, en la que, con la ayuda de los alumnos, se piense en un software que resuelva una problemática en particular con las herramientas aprendidas en el ejercicio (recibir información a través del teclado, transformaciones sobre los datos, impresión y dibujo en pantalla). El proyecto será completamente libre, para que puedan aplicar los aprendizajes construidos.

Algunos ejemplos:

1. Un programa que calcula si un año es bisiesto o no
2. Un programa que une varias frases *ingresadas a través del teclado* en una sola sentencia
3. Un programa que calcula si un número es primo o no
4. Un programa que realiza dibujos en base a distancias y ángulos de giro ingresados a través del teclado

Anexo

Cómo depurar?

Cuando existe algún error en nuestro código, el programa falla y no obtenemos los resultados esperados. Por suerte Python nos presenta con mensajes de error que nos ayudan a identificar y corregir los mismos. Por ejemplo, el siguiente código contiene un error

```
from time import *
from turtle import *

# Obtenemos los datos para personalizar la tarjeta
print 'Elige una amiga o amigo a quien conozcas hace muchos años.'
nombre=raw_input('¿Cual es su nombre? ')
comienzo_amistad=raw_inpu('¿En qué año lo conociste? (escribe las 4
cifras)')
ahora=strftime('%Y', gmtime()) # Guardamos el anio actual en una
variable
cantidad_anios=int(ahora)-int(comienzo_amistad) # Cálculo
```

Al ejecutarlo, Python nos presenta el siguiente mensaje de error:

```
============= RESTART: tarjeta_digital.py =============
Elige una amiga o amigo a quien conozcas hace muchos años.
¿Cual es su nombre? Felipe

Traceback (most recent call last):
  File "C:/Users/ROOT/Desktop/tarjeta_digital.py", line 7, in <module>
    comienzo_amistad=raw_inpu('¿En qué año lo conociste? (escribe las 4
cifras)')
NameError: name 'raw_inpu' is not defined
>>>
```

Lo primero que nos indica el mensaje es el nombre del archivo y la línea que contiene el error. En este caso “tarjeta_digital.py” linea 7.
Luego nos muestra el contenido de la línea que falló para finalmente dar una descripción del error. En este caso, que el nombre “raw_inpu” no está definido dentro de nuestro código. Esto es un error de sintaxis al escribir el código, uno de los errores más comunes de encontrar en la programación, y se resuelve escribiendo bien el código (en este caso “raw_input”) y ejecutando el código nuevamente.