

Programación

Números graficados

```
</> [ '0','1','0' ]  
{...}  
5 {  
6   aprender.a.programar;  
7   si situacion.problematica = verdadero  
8     repetir hasta que problema.resuelto = verdadero  
9     pienso.estrategia  
10    diseño.algoritmo  
11    busco.error (errores)  
12     si errores <> 0  
13     decir "Corrigiendo errores..."  
14     encuentro.error  
15     corrijo.error  
16     fin si  
17   fin repetir  
  fin si  
}
```

... { ^ _ ^ } ...

['1','1','0']

Actividad N° 9

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.
- El reconocimiento de cómo la información -en sus diversos formatos- es recolectada, representada, visualizada y analizada, a través de dispositivos computarizados, y la comprensión del uso de grandes volúmenes de datos, relacionados con la cuantificación, la predicción y la optimización de procesos, reflexionando sobre su utilidad social y sobre aspectos éticos vinculados al acceso a información de usuarios.

Objetivos de aprendizaje

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

- Introducir conceptos sobre el manejo de grandes volúmenes de datos (Big Data).
- Trabajar con Python como una herramienta para resolver problemas.
- Representar datos en forma visual para su mejor comprensión y análisis.
- Utilizar archivos de texto externos al programa para cargar datos.

Materiales y recursos

- Computadora
- Python 2.x instalado
- Módulo pygal para Python instalado. (Ver anexo al final del documento)



Desafío

En esta actividad utilizaremos Python para generar una representación visual a partir de un conjuntos de datos.

< Inicio >

Disparador

Así como las grandes entidades utilizan complejos algoritmos de análisis de datos para obtener información, nosotros podemos hacer lo mismo de forma muy simple con Python. El uso de gráficos permite visualizar conjuntos de datos de forma clara y simple, facilitando la comparación y el análisis. Python nos permite crear gráficos interactivos fácilmente. ¿Hacemos una encuesta en el colegio y analizamos de forma visual el resultado?

En esta actividad vamos a crear un programa que genera gráficos a partir de un conjunto de datos cargados desde un archivo de texto.

A grandes rasgos, el problema se descompone en:

- Importar librerías.
- Cargar los datos de un archivo de texto en un diccionario.
- Crear gráfico de tortas a partir de los datos.
- Crear gráfico de barras a partir de los datos.

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 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 hubiera.

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 ejecución de esta actividad se puede dividir en los siguientes pasos:

1. Realizar encuesta y guardar los resultados en un archivo de texto.
2. Importar librerías.
3. Cargar los datos desde el archivo de texto y guardarlos en un diccionario.
4. Crear gráfico de tortas a partir de los datos.
5. Crear gráfico de barras a partir de los datos.
6. Visualizar los gráficos.

El docente recuerda a los alumnos 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  
# Números Graficados
```

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

Se sugiere al docente hacer una pequeña introducción al concepto de Big Data para fomentar interés en la actividad.

Big data, o datos masivos, es un término que se utiliza para describir grandes cantidades de datos, que no pueden ser procesados o analizados utilizando procesos o herramientas tradicionales.

Un ejemplo de este tipo de conjunto de datos es la información creada por las personas a través del uso constante de computadoras, dispositivos móviles, reproductores de audio y video, cámaras fotográficas, sistemas GPS, sensores digitales, automóviles, redes sociales, y muchos otros elementos.

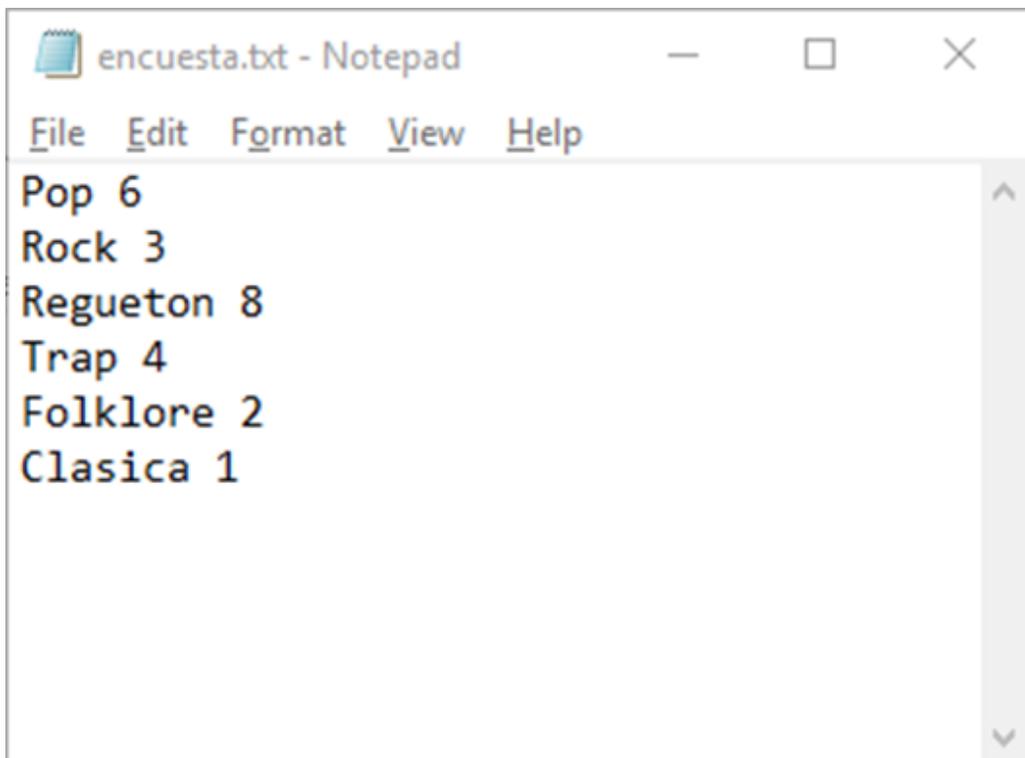
El análisis de Big Data se realiza para entender comportamientos y encontrar tendencias, es decir, puntos de referencia. De esta manera, las empresas u organismos que realizan el análisis saben qué acciones deben realizar para cumplir sus objetivos, anticipar y evitar problemas, reducir costos o aprovechar nuevas oportunidades.

1) Realizar encuesta y guardar los resultados en un archivo de texto.

Como ejemplo para el ejercicio de esta actividad se utiliza una encuesta sobre los gustos musicales de los estudiantes. Se sugiere al docente comenzar la actividad definiendo la temática de la encuesta junto a los estudiantes para fomentar la colaboración, el interés y la participación en la actividad.

Se sugiere al docente repasar conceptos sobre la representación visual de los datos y sus beneficios. A continuación se puede debatir sobre los beneficios y diferencias entre diferentes tipos de representación visual de datos (gráfico de tortas, de barras, lineal, etc.).

A continuación se realiza la encuesta definida y se guardan sus resultados en un archivo de nombre “encuesta.txt”. Por ejemplo:



```
encuesta.txt - Notepad
File Edit Format View Help
Pop 6
Rock 3
Regueton 8
Trap 4
Folklore 2
Clasica 1
```

Es necesario evitar el uso de acentos y caracteres especiales en el archivo de datos, o el programa fallará.

2) Importar librerías.

Como de costumbre, comenzamos importando las librerías que vamos a utilizar en nuestro programa. Importamos la librería “os”, que nos permitirá interactuar con el sistema operativo, y “pygal”, una librería de Python para crear gráficos a partir de datos.

```
# Importación de librerías
import pygal
import os
```

Si al ejecutar el código obtienes el siguiente error:

```
Traceback (most recent call last):
  File "graficos_encuesta.py", line 8, in <module>
    import pygal
ImportError: No module named pygal
```

dirigite a la sección Anexo al final de este documento, y sigue las instrucciones para instalar el módulo “pygal” en tu computadora.

3) Cargar los datos desde el archivo de texto y guardarlos en un diccionario.

Trabajar con archivos de texto en Python nos permite separar los datos del código. De esta manera podemos escribir un mismo código que resuelva de forma genérica una problemática. Esto significa que podemos utilizar el mismo programa para trabajar con diferentes conjuntos de datos.

Para esta actividad, debemos cargar los datos de la encuesta previamente guardados en el archivo “encuesta.txt”. Utilizaremos la función “open()”, que nos permite abrir un archivo y trabajar con su contenido. Al trabajar con archivos externos, es importante contemplar en nuestro código los posibles errores que esto conlleva, como que no exista el archivo que intentamos abrir, que no tengamos los permisos suficientes o acordarnos de cerrar el archivo al finalizar su utilización. Por suerte Python cuenta con la palabra clave “with”. Al abrir un archivo con “with”, Python se encarga de considerar los posibles escenarios de error y de cerrar el archivo, una vez terminamos de trabajar con él.

Una vez abierto el archivo, creamos un bucle “for” para recorrerlo línea por línea. Finalmente utilizamos la función “split()” para separar el contenido de la línea en distintas variables y guardarlo dentro de un diccionario como llaves y valores. A continuación el código con comentarios para su mejor comprensión.

```

# Creamos el diccionario resultados donde guardaremos nuestros datos.
resultados = {}
# Abrimos el archivo con open(),
# y utilizamos with para manejar los posibles errores.
with open("encuesta.txt") as archivo:
    # Recorremos el archivo línea a línea
    for linea in archivo:
        # Separamos el contenido de la línea en variables, llave y valor.
        (llave, valor) = linea.split()
        # Por último guardamos los valores obtenidos en el diccionario.
        resultados[llave] = int(valor)

# Imprimimos en pantalla el contenido del diccionario
print resultados
{'Regueton': 8, 'Clasica': 1, 'Pop': 6, 'Trap': 4, 'Folklore': 2,
'Rock': 3}

```

*¿Notaste que utilizamos la función int para guardar el valor en el diccionario?
¿Podrías explicar por qué?*

4) Crear gráfico de tortas a partir de los datos.

La librería “pygal” nos permite crear todo tipo de gráficos a partir de un conjunto de datos. Comenzaremos creando un nuevo gráfico de tortas “grafico torta” utilizando la función “pygal.Pie()”. Le damos un título a nuestro gráfico, asignando un valor a la propiedad “title” de “grafico torta” y utilizamos un loop “for” para cargar los datos de nuestro diccionario utilizando la función “add()” de “grafico torta”. Como último generamos la imagen resultado y la guardamos en la carpeta “Imágenes”.

```

# Creación del gráfico de torta.
grafico_torta = pygal.Pie()
# Asignación de un título al grafico.
grafico_torta.title = "Estilos musicales preferidos."
# Carga de datos al gráfico.
for llave in resultados:
    grafico_torta.add(llave, resultados[llave])
# Generación del archivo de imagen.
grafico_torta.render_to_file(os.getcwd()+ '\Imágenes\grafico_torta.svg')

```

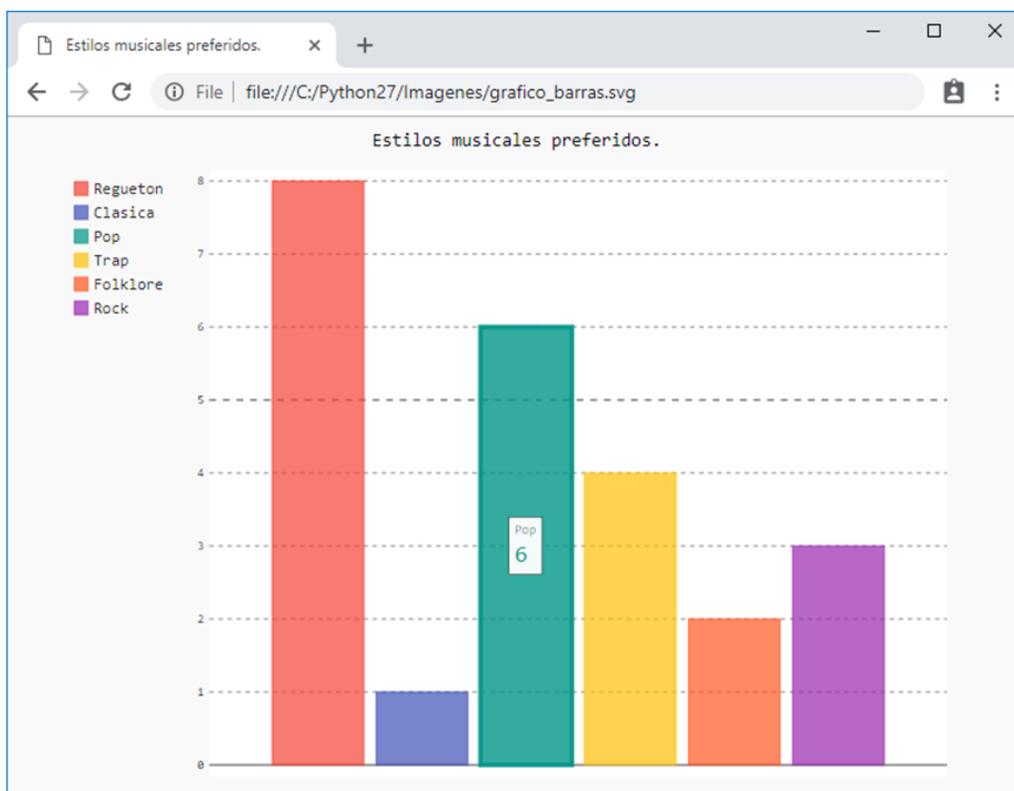
5) Crear gráfico de barras a partir de los datos.

La librería “pygal” nos permite crear muchos tipos de gráficos, no solo de tortas. Repetimos los pasos del punto anterior para crear un nuevo gráfico, estas vez de barras.

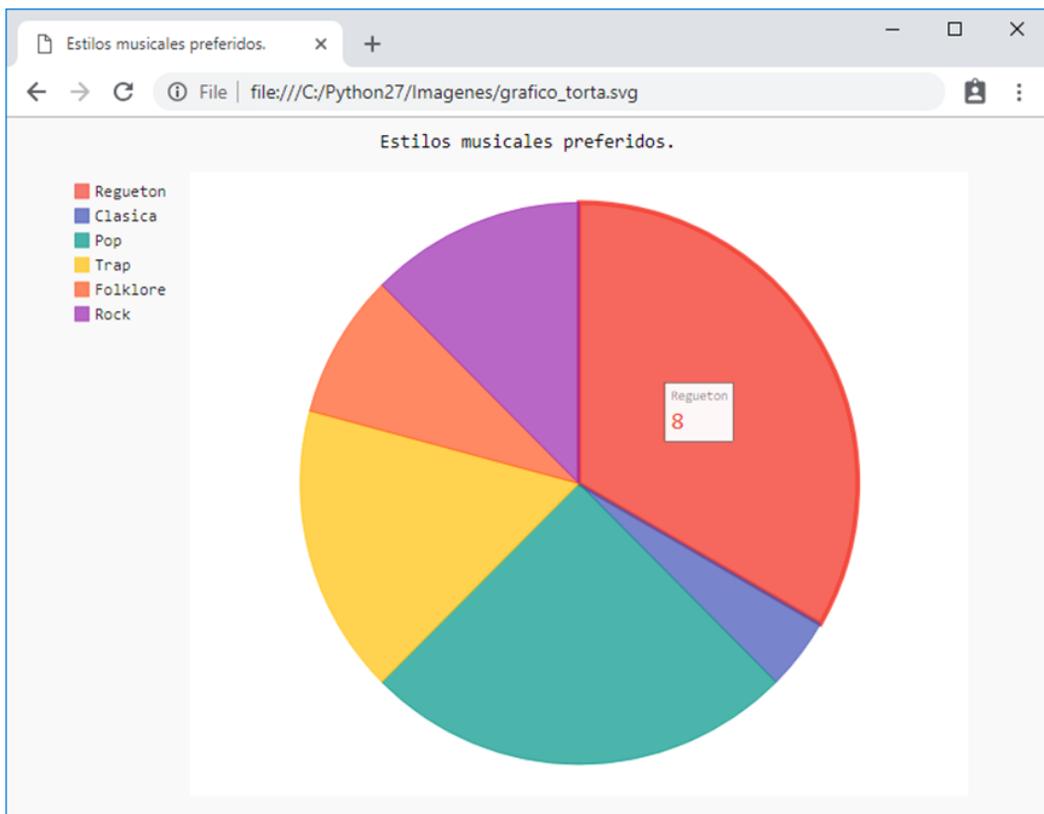
```
# Creación del gráfico de barras.
grafico_barras = pygal.Bar()
# Asignación de un título al grafico.
grafico_barras.title = "Estilos musicales preferidos."
# Carga de datos al gráfico.
for llave in resultados:
    grafico_barras.add(llave, resultados[llave])
# Generación del archivo de imagen.
grafico_barras.render_to_file(os.getcwd()+
'\Imágenes\grafico_barras.svg')
```

6) Visualizar los gráficos.

Los gráficos generados por nuestro programa se encuentran en la carpeta “Imágenes” de nuestra área de trabajo. Podemos abrirlos con cualquier navegador web. Una vez abiertos, mové el *mouse* sobre los gráficos para ver información extra. También podés hacer clic sobre las categorías que están a la izquierda, para activar o desactivar su visualización en el gráfico.



Ventana ejemplo de gráfico de barras interactivo



Ventana ejemplo de gráfico de torta interactivo

Código completo

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

```
# -*- coding: cp1252 -*-
# -*- coding: 850 -*-
# -*- coding: utf-8 -*-
# Creador: Pedro Perez
# Números Graficados

# Importación de librerías
import pygal
import os

# Cargamos los datos desde el archivo de texto
# y los guardamos en el diccionario "resultados"
resultados = {}
with open("encuesta.txt") as archivo:
    for linea in archivo:
        (llave, valor) = linea.split()
        resultados[llave] = int(valor)

# print resultados
```

```

# Creación del gráfico de tortas.
grafico_torta = pygal.Pie()
grafico_torta.title = "Estilos musicales preferidos."
# Carga de datos al gráfico.
for llave in resultados:
    grafico_torta.add(llave, resultados[llave])
# Generación del archivo de imagen.
grafico_torta.render_to_file(os.getcwd()+
'\Imágenes\grafico_torta.svg')

# Creación del gráfico de barras.
grafico_barras = pygal.Bar()
grafico_barras.title = "Estilos musicales preferidos."
# Carga de datos al gráfico.
for llave in resultados:
    grafico_barras.add(llave, resultados[llave])
# Generación del archivo de imagen.
grafico_barras.render_to_file(os.getcwd()+
'\Imágenes\grafico_barras.svg')

```

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

Los gráficos serán creados dentro de la carpeta “Imágenes” en tu directorio de trabajo.

< Cierre >

El docente pide a los estudiantes que se agrupen de a dos, 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.

Evaluación

El docente puede evaluar el proyecto tanto a través de la observación, durante el desarrollo de las actividades, como en relación al programa final, para lo cual podrá acercarse a cada alumno, o bien, si se quisiese ver en detalle el código, se podrán copiar los archivos con los programas desarrollados.

Se tendrán en cuenta los objetivos específicos de la actividad, como otros aspectos vinculados a la creatividad, la cooperación entre pares y el aprendizaje a partir de la exploración y el error. Para evaluar esto, se le puede pedir a cada alumno que explique qué desafíos encontró y cómo los solucionó.

A continuación, se presentan preguntas orientadoras:

- ¿Logró resolver el desafío propuesto?
- ¿Pudo cargar los datos desde el archivo y crear la estructura de datos sin dificultades?
- ¿Trabajó en grupo cuando se encontró con problemas?
- ¿Escribió el programas propuesto de forma lógica y ordenada, solucionando los errores de sintaxis, de haberse presentado?
- ¿Modificó de alguna forma su código para mejorar o modificar el del ejemplo de la actividad?

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

Para seguir aprendiendo

Para continuar sugerimos al docente realizar una modificación al código de la actividad 3: Trivia Somos Digitales utilizando las herramientas aprendidas durante esta actividad para extender la funcionalidad del programa, generando un gráfico de barras que compara la cantidad de respuestas correctas versus la cantidad de respuestas incorrectas.

Otro posible ejercicio es la modificación del cifrador y descifrador creados en la actividad 8 para que ambos programas carguen los caracteres del alfabeto desde un archivo. De esta manera solo hay que cambiar el archivo de texto al modificar o extender el alfabeto facilitando el mantenimiento del alfabeto.

Anexo

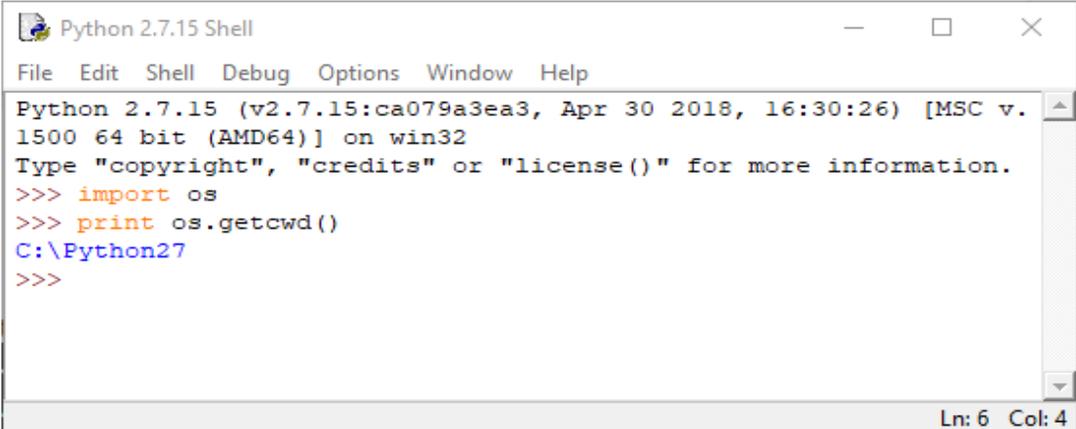
Como instalar el módulo pygal en Python

Si al ejecutar el programa encontrás el siguiente error:

```
Traceback (most recent call last):
  File "graficos_encuesta.py", line 8, in <module>
    import pygal
ImportError: No module named pygal
```

Significa que no tenés instalado el módulo de Python “pygal” en tu computadora.

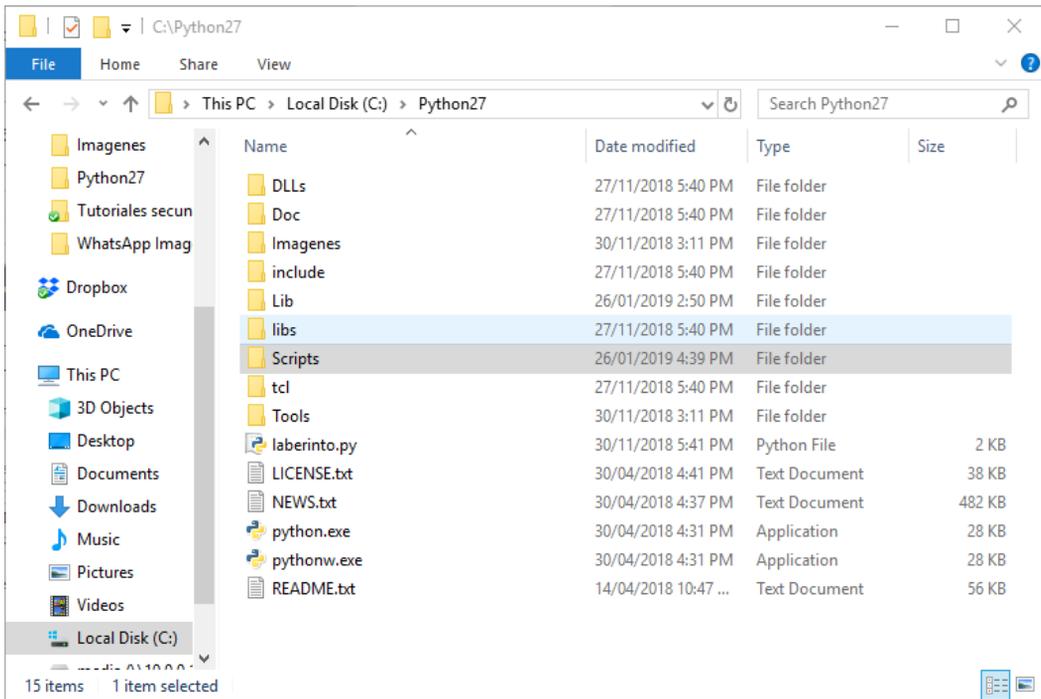
Para instalarlo, primero debemos averiguar en qué directorio se encuentra instalado Python. Para esto, podés ejecutar las siguientes líneas desde la consola IDLE de Python:



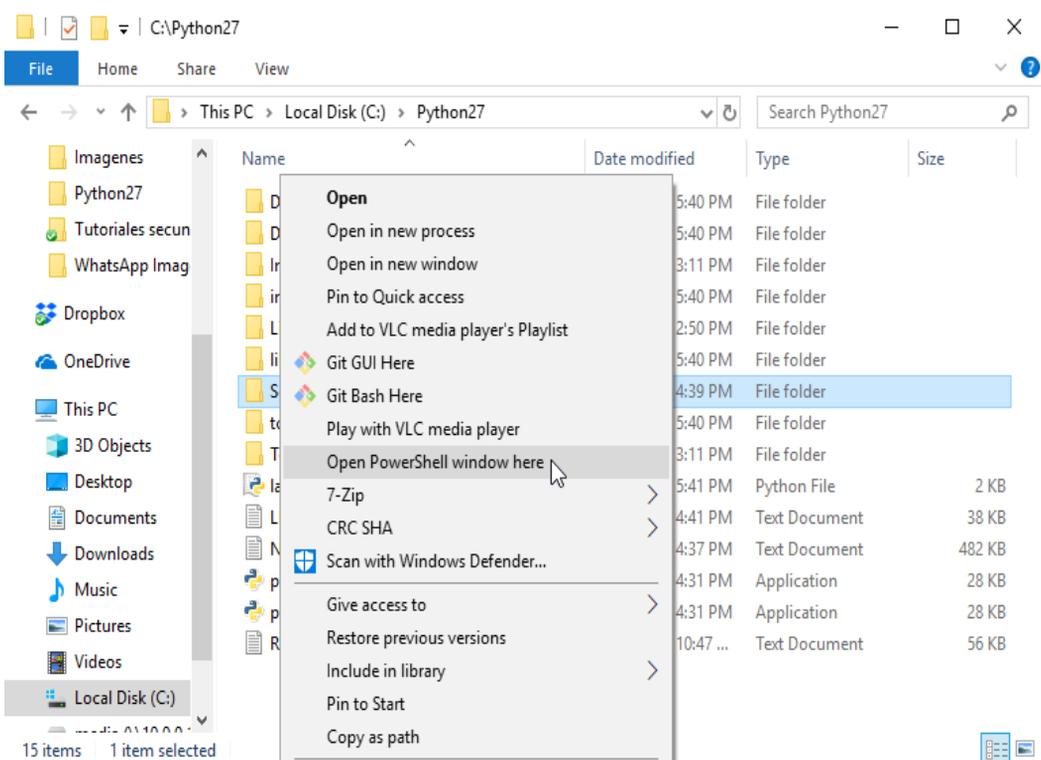
```
Python 2.7.15 Shell
File Edit Shell Debug Options Window Help
Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.
1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import os
>>> print os.getcwd()
C:\Python27
>>>
```

En nuestro ejemplo, Python se encuentra instalado en el directorio C:\Python27.

Nos posicionamos dentro de ese directorio desde el navegador de archivos, e ingresamos a la carpeta “Scripts”:



Una vez dentro de la carpeta Scripts, presionamos el botón derecho del *mouse* a la vez que mantenemos presionada la tecla “Shift”. De esta manera se despliega el menú contextual avanzado. Seleccionamos la opción “Abrir ventana de comandos aquí”:



Una vez que tenemos la ventana de comandos abierta, ejecutamos la siguiente línea para instalar el módulo “pygal” en nuestro sistema.

```
PS C:\Python27\Scripts> .\pip install pygal
Collecting pygal
  https://files.pythonhosted.org/packages/5f/b7/201c9254ac0d2b8ffa3bb2d528
d23a4130876d9ba90bc28e99633f323f17/pygal-2.4.0-py2.py3-none-any.whl
Installing collected packages: pygal
Successfully installed pygal-2.4.0
```

Ahora podrás ejecutar el programa sin errores.