Colección de actividades Aprender Conectados

Nivel Secundario

Programación

Somos digitales

Actividad N° 3

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:

- Trabajar con entrada y salida de datos en tiempo real.
- Conocer las estructuras de datos “lista” y “diccionario”.
- Tomar decisiones mediante estructuras de selección.
- Iterar repeticiones de código utilizando bucles.
- Crear un juego de trivia que reciba respuestas por teclado y evalúe si las respuestas son correctas o no.

Materiales y recursos

- Computadora.
- Python 2.x instalado.

Desafío

En esta actividad creamos un juego de preguntas y respuestas sobre la serie Somos Digitales. Para esto nos valdremos de estructuras de datos, estructuras de selección y bucles. El usuario debe ingresar las respuestas por teclado y el programa evaluará si la respuesta es correcta o no.

< Inicio >

Disparador.
¿Están listos para responder algunas preguntas sobre el capítulo 2 de Somos Digitales?

En esta actividad debemos crear un programa de preguntas y respuestas personalizadas. El desafío, que presenta cierta complejidad, puede ser resuelto fácilmente por los alumnos con ayuda del docente y mediante la descomposición del problema en partes.

A grandes rasgos, el problema se descompone en:

- Definir las preguntas y sus posibles respuestas.
- Crear la estructura de datos que contenga las preguntas, las respuestas y cuál es la correcta.
- Imprimir un mensaje con las instrucciones de uso.
- Crear un bucle que recorra la lista de preguntas e imprima en pantalla las preguntas con sus posibles respuestas.
- Crear una estructura de selección que evalúe si la respuesta del usuario es la correcta, y lo notifique.

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.

Se sugiere personalizar el ejercicio de manera que los alumnos creen sus propias preguntas y respuestas.

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

- Definir las preguntas y sus posibles respuestas.
- Crear la estructura de datos que contenga las preguntas, las respuestas y cuál es la correcta.
- Imprimir un mensaje con las instrucciones de uso.
- Crear un bucle que recorra la lista de preguntas e imprima en pantalla las preguntas con sus posibles respuestas.
- Crear una estructura de selección que evalúe si la respuesta del usuario es la correcta, y lo notifique.

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
# Trivia Somos Digitales
```

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

1) Definir las preguntas y sus posibles respuestas.

Antes de empezar a programar hay que definir las preguntas y sus respuestas. Se sugiere proponer a los alumnos la creación de trivias personalizadas. Para el ejercicio de ejemplo vamos a proponer algunas preguntas sobre el capítulo 2 de la serie Somos Digitales.

Las preguntas propuestas con sus respuestas son:

- ¿Cuál es la clave para estar conectado?
  - H4x0rRu73Z
  - ciberespacio
  - **C1berespac1O**
  - mama123
- ¿Qué es internet?
  - Un dispositivo que nos conecta al wifi.
  - **Una red global que conecta a millones de dispositivos entre sí.**
  - Un espacio digital donde compartir información con otras personas.
  - Un plato típico de la india.
- ¿Cuál es la diferencia entre las redes fijas y las mobiles?
  - **Las redes fijas son más estables pero su cobertura está definida en un área específica, como una oficina, mientras que las redes móviles nos permiten seguir conectados en tránsito, incluso en movimiento por largas distancias.**
  - A las redes fijas se accede mediante un medio físico, como un cable o una fibra óptica, mientras que a las redes móviles se accede a través de una conexión inalámbrica como WiFi o 4G.
  - Las redes fijas son redes de poliéster que se mantienen fijadas al suelo, mientras que las redes móviles son redes de poliéster sujetas a ruedas para ser transportadas.

    - Las redes fijas son las redes que siempre se encuentran en un mismo lugar, como una oficina, mientras que las redes móviles son redes portátiles, como un vehículo con wifi que puede llevarse de un lugar a otro.
- ¿Donde se alojan las páginas a las que accedemos a través de la Word Wide Web (WWW)?
    - En el aire.
    - En las nubes.
    - En la estratosfera.
    - **En servidores informáticos distribuidos por todo el mundo.**
- ¿Cómo se denominan los programas que nos permiten ingresar a la Word Wide Web (WWW)?
    - Exploradores.
    - Navegantes.
    - **Navegadores.**
    - Aventureros.
- La nube nos permite acceder a información y servicios sin necesidad de tenerlos descargados en nuestros dispositivos. Pero hay un requisito esencial para acceder a la nube. ¿Cual es?
    - **Contar con acceso a Internet.**
    - Estar conectados a una red.
    - Tener espacio disponible en el dispositivo.
    - Almorzar liviano y sin gaseosa.
- ¿Qué es el ciberespacio?
    - Es otra forma de denominar a Internet.
    - **Un espacio virtual donde compartimos e intercambiamos todo tipo de información.**
    - Lo que hay más allá del espacio.
    - Un espacio de encuentro para robots.
- Cada vez que navegamos, compramos, compartimos y descargamos información por internet, estamos generando datos. Esta inmensa cantidad de datos se guardan para su posterior análisis y explotación. ¿Como se denomina a las grandes cantidades de datos en informática?
    - Base de datos.
    - Ciberespacio.
    - **Big Data.**
    - Internet.

2) Crear la estructura de datos que contenga las preguntas, las respuestas y cuál es la correcta.

Una vez definidas las preguntas, debemos cargarlas en nuestro programa. Si bien podemos guardar las preguntas en simples variables, nos encontramos con el subsiguiente problema de relacionar las preguntas con sus posibles respuestas y de identificar cual es la correcta entre ellas. Una forma elegante y simple de resolver este problema es con la utilización de estructuras de datos. En ciencias de la computación, una estructura de datos es una forma particular de organizar datos en una computadora para que puedan ser utilizados de manera eficiente. Para resolver este problema utilizaremos dos estructuras de datos propias de Python llamadas listas y diccionario de datos.
Las listas son listados de datos en los que hay un orden, por lo que se tiene en cuenta la posición en la que está el elemento. Recuerda que el primer elemento es el número 0, y no el número 1. En las listas se pueden modificar sus elementos, y puede haber elementos duplicados. Se crean poniendo sus elementos entre corchetes "[]" y separados por comas ",", como por ejemplo:

```
# Definimos la lista de compras con sus elementos
lista_compras = ["Leche", "Galletas", "Miel", "Agua", "Servilletas"]
# El orden comienza con el número 0, por lo que si queremos modificar
# el ítem "Miel", lo haremos usando el índice 2 (tercera posición)
lista_compras[2] = "Mermelada"
# Podemos ver la nueva lista con print:
print lista_compras
['Leche', 'Galletas', 'Mermelada', 'Agua', 'Servilletas']
```

Los elementos de una lista pueden ser cualquier tipo de datos, como números, texto o incluso otras estructuras de datos, como listas y diccionarios. Se te ocurre para que se puede utilizar una lista de listas?

Ahora que conocemos las listas podemos resolver el problema de ordenar nuestras preguntas y sus respuestas en listas de datos. Lo que las listas no nos permiten resolver, es cómo asociar las preguntas a sus respuestas y cómo identificar la correcta entre ellas. Para esto utilizamos otra estructura de datos llamada diccionario de datos.
Como sucede con un diccionario convencional, un diccionario en Python es una palabra que tiene un elemento asociado. Al contrario de lo que sucedía en las listas, los elementos del diccionarios no tienen orden. Se denominan llaves o keys a las "palabras clave" y valores o values a las "definiciones". Lógicamente, no puede haber dos llaves iguales, aunque sí dos valores iguales. Los diccionarios se crean poniendo sus elementos entre llaves "{}", como por ejemplo:

```
capitales = {"Argentina":"Buenos Aires", "Brasil":"Brasília",
"Uruguay":"Montevideo"}
```

Como los diccionarios no tiene orden, se puede acceder o modificar sus elementos utilizando la llave como índice. Por ejemplo:

```
capitales = {"Argentina":"Buenos Aires", "Brasil":"Brasília",
"Uruguay":"Montevideo"}

capitales["Argentina"] = "Bs. As."

print capitales
{'Argentina': 'Bs. As.', 'Brasil': 'Brasília', 'Uruguay': 'Montevideo'}
```

Ahora sí, combinando estas estructuras de datos podemos resolver nuestra problemática y definir las preguntas y sus respuestas como datos estructurados. Para esto creamos una lista de preguntas, donde cada elemento de la lista es un diccionario que tiene tres llaves con sus correspondientes valores: pregunta, opciones y respuesta.

```
# Definimos la lista preguntas.
# Reconocemos que es una lista porque abre con "["
 preguntas = [
  # Dentro de la lista, cada elemento es un diccionario.
  # Lo reconocemos dado que abre con "{"
  {
  # El valor de la llave 'pregunta' es la pregunta
  'pregunta': '¿Cuál es la clave para estar conectado?',
  # El valor de la llave 'opciones', es una lista de posibles
respuestas
  'opciones': ['H4x0rRu73Z', 'ciberespacio', 'C1berespac1O',
'mama123'],
  # El valor de la llave 'respuesta' es la posición de la respuesta
  # correcta dentro de la lista definida en la llave 'opciones'
  'respuesta': 3
  },

  # Creamos el resto de las preguntas siguiendo la misma estructura
  {
  'pregunta': '¿Qué es internet?',
  'opciones': ['Un dispositivo que nos conecta al wifi.', 'Una red
global que conecta a millones de dispositivos entre sí.', 'Un
espacio digital donde compartir información con otras personas.',
'Un plato típico de la india.'],
  'respuesta': 2
```

```
  },

  {
  'pregunta': '¿Cuál es la diferencia entre las redes fijas y las
mobiles?',
  'opciones': ['Las redes fijas son más estables pero su cobertura
está definida en un área específica, como una oficina, mientras que
las redes móviles nos permiten seguir conectados en tránsito,
incluso en movimiento por largas distancias.', 'A las redes fijas
se accede mediante un medio físico, como un cable o una fibra
óptica, mientras que a las redes móviles se accede a través de una
conexión inalámbrica como WiFi o 4G.', 'Las redes fijas son redes
de poliéster que se mantienen fijadas al suelo, mientras que las
redes móviles son redes de poliéster sujetas a ruedas para ser
transportadas.', 'Las redes fijas son las redes que siempre se
encuentran en un mismo lugar, como una oficina, mientras que las
redes móviles son redes portátiles, como un vehículo con wifi que
puede llevarse de un lugar a otro.'],
  'respuesta': 1
  },

  {
  'pregunta': '¿Donde se alojan las páginas a las que accedemos a
través de la Word Wide Web (WWW)?',
  'opciones': ['En el aire.', 'En las nubes.', 'En la
estratosfera.', 'En servidores informáticos distribuidos por todo
el mundo.'],
  'respuesta': 4
  },

  {
  'pregunta': '¿Cómo se denominan los programas que nos permiten
ingresar a la Word Wide Web (WWW)?',
  'opciones': ['Exploradores.', 'Navegantes.', 'Navegadores.',
'Aventureros.'],
  'respuesta': 3
  },

  {
   'pregunta': 'La nube nos permite acceder a información y
servicios sin necesidad de tenerlos descargados en nuestros
dispositivos. Pero hay un requisito esencial para acceder a la
nube. ¿Cual es?',
   'opciones': ['Contar con acceso a Internet.', 'Estar conectados
a una red.', 'Tener espacio disponible en el dispositivo.',
'Almorzar liviano y sin gaseosa.'],
   'respuesta': 1
  },
```

```
  {
  'pregunta': '¿Qué es el ciberespacio?',
  'opciones': ['Es otra forma de denominar a Internet.', 'Un
espacio virtual donde compartimos e intercambiamos todo tipo de
información.', 'Lo que hay más allá del espacio.', 'Un espacio de
encuentro para robots.'],
  'respuesta': 2
  },

  {
  'pregunta': 'Cada vez que navegamos, compramos, compartimos y
descargamos información por internet, estamos generando datos. Esta
inmensa cantidad de datos se guardan para su posterior análisis y
explotación. ¿Como se denomina a las grandes cantidades de datos en
informática?',
  'opciones': ['Base de datos.', 'Ciberespacio.', 'Big Data.',
'Internet.'],
  'respuesta': 3
  }
  ] # Cerramos la lista de diccionarios
```

3) Imprimir un mensaje con las instrucciones de uso.

Antes de definir el algoritmo que presenta las preguntas y evalúa las respuestas del usuario, es necesario presentar al usuario una explicación del objetivo y como jugarlo. Lo hacemos utilizando la opción print.

```
print "################################################"
print "## Bienvenido a la trivia 'Somos Digitales' ##"
print "################################################"
print
print "Para ganar debes elegir la respuesta correcta a las preguntas."
print "Ingresa por teclado el número de tu respuesta."
print "¡Mucha suerte!"
print
print
```

¿Se te ocurre cómo imprimir todo este mensaje en una sola línea de print?

4) Crear un bucle que recorra la lista de preguntas e imprima en pantalla las preguntas con sus posibles respuestas.

Ahora tenemos que recorrer la lista de preguntas e imprimirlas en pantalla junto a sus posibles respuestas para que el usuario pueda elegir su respuesta. Para esto vamos a usar un bucle del tipo “for”. Un bucle es una estructura de control que repite un bloque de código mientras se cumpla una condición dada. El bloque de código que se repite se llama cuerpo del bucle y cada repetición se llama iteración.

Para el siguiente ejemplo el bucle “for” recorre la lista de ítems, y en cada iteración asigna el valor del ítem a la variable “item”. De esta manera nos permite trabajar dentro del cuerpo del bucle con los diferentes valores de la lista. En este ejemplo imprimimos su contenido por pantalla:

```
for item in ["item 1", "item 2", "otro ítem"]:
  print item # Cuerpo del bucle. Nótese la indentación.
item 1
item 2
otro ítem
```

Para resolver nuestro desafío vamos a recorrer la lista de preguntas con un bucle “for” y vamos a imprimir los valores de las llaves “pregunta” y “opciones”. Dado que el valor de “opciones” es otra lista, utilizamos otro bucle “for” dentro del bucle principal para recorrer e imprimir cada una de las opciones. Cuando tenemos bucles dentro de bucles, se lo llama bucles anidados.

```
# Recorremos los valores de la lista de preguntas una por una.
# El bucle "for" asigna el contenido a la variable "item".
for item in preguntas:
  # Imprimimos el valor de la llave "pregunta" en el cuerpo del bucle.
  # Esta acción se repetirá por cada ítem dentro de la lista.
  print item['pregunta']
  # A continuación creamos una variable con el valor 1.
  # Utilizaremos esta variable para asignar un número a cada opción.
  # El usuario utilizará este número para indicar su respuesta.
  numero = 1
  # Creamos un bucle "for" anidado para recorrer e imprimir
  # cada ítem de la lista "opciones".
  for opcion in item['opciones']:
    print str(numero) +") " + opcion
    numero = numero + 1
```

¿Puedes explicar por qué utilizamos la función “srt()” para imprimir el numero de opcion?

¿Se te ocurre otra manera de integrar el número a las opciones?

5) Crear una estructura de selección que evalúe si la respuesta del usuario es la correcta, y lo notifique.

El último paso de nuestro desafío es recibir la respuesta del usuario a través del teclado y evaluar si el número de opción elegido es el correcto. Para esto utilizaremos una estructura de selección. Todo lenguaje de programación moderno cuenta con estas estructuras que nos permiten evaluar una condición y ejecutar o no un bloque de código en base al resultado. La estructura de selección simple se llama “if”, y a continuación vemos un ejemplo de su sintaxis:

```
nota = 7 # Definimos el valor 7 a la variable "nota".
if nota >= 6: # Evaluamos si la nota es mayor o igual a 6.
    print "Aprobado" # Si cumple la condición, ejecuta el código.
```

¿Puedes indicar el resultado de ejecutar el código de ejemplo?

¿Pero qué sucede si necesitamos ejecutar un bloque de código en el caso de que la condición no se cumpla? Para esto existe la sentencia “else”. Esta nos permite ejecutar el bloque de codigo a continuacion en caso de que la sentencia “if” anterior falle. Ejemplo:

```
nota = 5
if nota >= 6:
    print "Aprobado"
else: # Si el valor de "nota" NO es mayor o igual a 6,
      # ejecutar el siguiente bloque de código.
    print "Reprobado"
```

¿Puedes indicar el resultado de ejecutar el código de ejemplo?

Para resolver nuestro desafío, primero vamos a recibir la respuesta del usuario mediante la función “raw_input()”, para luego comparar su respuesta con el valor de la llave “respuesta” del correspondiente diccionario. Dado que estas acciones son realizadas dentro del bucle que recorre las preguntas, presentamos el bucle completo en el código de ejemplo para su mayor claridad.

```
# Recorremos los valores de la lista de preguntas una por una.
# El bucle "for" asigna el contenido a la variable "item".
for item in preguntas:
  # Imprimimos el valor de la llave "pregunta" en el cuerpo del bucle.
  # Esta acción se repetirá por cada ítem dentro de la lista.
  print item['pregunta']
  # A continuación creamos una variable con el valor 1.
  # Utilizaremos esta variable para asignar un número a cada opción.
  # El usuario utilizará este número para indicar su respuesta.
  numero = 1
  # Creamos un bucle "for" anidado para recorrer e imprimir
  # cada ítem de la lista "opciones".
  for opcion in item['opciones']:
    print str(numero) +") " + opcion
    numero = numero + 1
  # Recibimos la respuesta del usuario por teclado.
  # Notese el cambio de indentación.
  # Esto indica que salimos del bucle de opciones, y volvimos al bucle
principal
```

```
  respuesta = raw_input('Respuesta: ')
  # Evaluamos si la respuesta es correcta
  if int(respuesta) == item['respuesta']:
    # Informamos si la respuesta fue correcta.
    print "CORRECTO \n"
  else:
    # Informamos si la respuesta fue incorrecta.
    print "equivocado \n"
  print
```

¿Se te ocurre porque se usan dos símbolos "=" en vez de uno para realizar la comparación?

Código completo

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

```
# -*- coding: cp1252 -*-
# -*- coding: 850 -*-
# -*- coding: utf-8 -*-
# Creado por Pedro Pérez
# Trivia Somos Digitales

# Definimos la lista preguntas.
# Reconocemos que es una lista porque abre con "["
preguntas = [
  # Dentro de la lista, cada elemento es un diccionario.
  # Lo reconocemos dado que abre con "{"
  {
  # El valor de la llave 'pregunta' es la pregunta
  'pregunta': '¿Cuál es la clave para estar conectado?',
  # El valor de la llave 'opciones', es una lista de posibles respuestas
  'opciones': ['H4x0rRu73Z', 'ciberespacio', 'C1berespac1O', 'mama123'],
  # El valor de la llave 'respuesta' es la posición de la respuesta
  # correcta dentro de la lista definida en la llave 'opciones'
  'respuesta': 3
  },

  # Creamos el resto de las preguntas siguiendo la misma estructura
  {
  'pregunta': '¿Qué es internet?',
  'opciones': ['Un dispositivo que nos conecta al wifi.', 'Una red
global que conecta a millones de dispositivos entre sí.', 'Un espacio
digital donde compartir información con otras personas.', 'Un plato
típico de la india.'],
  'respuesta': 2
```

```
  },

  {
  'pregunta': '¿Cuál es la diferencia entre las redes fijas y las
mobiles?',
  'opciones': ['Las redes fijas son más estables pero su cobertura está
definida en un área específica, como una oficina, mientras que las redes
móviles nos permiten seguir conectados en tránsito, incluso en
movimiento por largas distancias.', 'A las redes fijas se accede
mediante un medio físico, como un cable o una fibra óptica, mientras que
a las redes móviles se accede a través de una conexión inalámbrica como
WiFi o 4G.', 'Las redes fijas son redes de poliéster que se mantienen
fijadas al suelo, mientras que las redes móviles son redes de poliéster
sujetas a ruedas para ser transportadas.', 'Las redes fijas son las
redes que siempre se encuentran en un mismo lugar, como una oficina,
mientras que las redes móviles son redes portátiles, como un vehículo
con wifi que puede llevarse de un lugar a otro.'],
  },

  {
  'pregunta': '¿Donde se alojan las páginas a las que accedemos a través
de la Word Wide Web (WWW)?',
  'opciones': ['En el aire.', 'En las nubes.', 'En la estratosfera.', 'En
servidores informáticos distribuidos por todo el mundo.'],
  'respuesta': 4
  },

  {
  'pregunta': '¿Cómo se denominan los programas que nos permiten ingresar
a la Word Wide Web (WWW)?',
  'opciones': ['Exploradores.', 'Navegantes.', 'Navegadores.',
'Aventureros.'],
  'respuesta': 3
  },

  {
   'pregunta': 'La nube nos permite acceder a información y servicios sin
necesidad de tenerlos descargados en nuestros dispositivos. Pero hay un
requisito esencial para acceder a la nube. ¿Cual es?',
   'opciones': ['Contar con acceso a Internet.', 'Estar conectados a una
red.', 'Tener espacio disponible en el dispositivo.', 'Almorzar liviano y
sin gaseosa.'],
   'respuesta': 1
```

```
 },

  {
  'pregunta': '¿Qué es el ciberespacio?',
  'opciones': ['Es otra forma de denominar a Internet.', 'Un
espacio virtual donde compartimos e intercambiamos todo tipo de
información.', 'Lo que hay más allá del espacio.', 'Un espacio de
encuentro para robots.'],
  'respuesta': 2
  },

  {
  'pregunta': 'Cada vez que navegamos, compramos, compartimos y
descargamos información por internet, estamos generando datos.
Esta inmensa cantidad de datos se guardan para su posterior
análisis y explotación. ¿Como se denomina a las grandes cantidades
de datos en informática?',
  'opciones': ['Base de datos.', 'Ciberespacio.', 'Big Data.',
'Internet.'],
  'respuesta': 3
  }
  ] # Cerramos la lista de diccionarios


print "################################################"
print "## Bienvenido a la trivia 'Somos Digitales' ##"
print "################################################"
print
print "Para ganar debes elegir la respuesta correcta a las
preguntas."
print "Ingresa por teclado el número de tu respuesta."
print "¡Mucha suerte!"
print
print


# Recorremos los valores de la lista de preguntas una por una.
# El bucle "for" asigna el contenido a la variable "item".
for item in preguntas:
  # Imprimimos el valor de la llave "pregunta" en el cuerpo del
bucle.
  # Esta acción se repetirá por cada ítem dentro de la lista.
  print item['pregunta']
  # A continuación creamos una variable con el valor 1.
  # Utilizaremos esta variable para asignar un número a cada
opción.
  # El usuario utilizara este número para indicar su respuesta.
  numero = 1
```

```
  # Creamos un bucle "for" anidado para recorrer e imprimir
  # cada ítem de la lista "opciones".
  for opcion in item['opciones']:
    print str(numero) +") " + opcion
    numero = numero + 1
  # Recibimos la respuesta del usuario por teclado.
  # Nótese el cambio de indentación.
  # Esto indica que salimos del bucle de opciones, y volvimos al
bucle principal
  respuesta = raw_input('Respuesta: ')
  # Evaluamos si la respuesta es correcta
  if int(respuesta) == item['respuesta']:
    # Informamos si la respuesta fue correcta.
    print "CORRECTO \n"
  else:
    # Informamos si la respuesta fue incorrecta.
    print "equivocado \n"
  print
```

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

< 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 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?
- ¿Comprendió la lógica del bucle y la estructura de selección?

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 plantear una actividad en grupo, en la que -con la ayuda de los estudiantes- se piense en un programa que resuelva una problemática en particular y que pueda ser resuelto con las herramientas aprendidas en el ejercicio (diccionarios, bucles y estructuras de selección).

El proyecto será completamente libre, para que puedan aplicar los aprendizajes construidos.

Algunos ejemplos:

- Un juego estilo elige tu propia aventura, donde la historia se desarrolla en base a las elecciones del usuario.
- Un juego para adivinar un número secreto. El usuario debe ingresar el número que cree es el correcto, y el programa le tiene que decir si es correcto, si es menor o si mayor. El usuario puede continuar preguntando hasta llegar al número secreto.

Anexo

A continuación se presenta en mayor profundidad la utilización de listas y diccionarios en Python.

Operaciones más habituales con listas en Python

Las operaciones más habituales que se realizan en Python son las siguientes:

- `lista[i]:` Devuelve el elemento que está en la posición i de la lista.
- `lista.pop(i):` Devuelve el elemento en la posición i de una lista y luego lo borra.

- `lista.append(elemento)`: Añade elemento al final de la lista.
- `lista.insert(i, elemento)`: Inserta elemento en la posición i.
- `lista.extend(lista2)`: Fusiona lista con lista2.
- `lista.remove(elemento)`: Elimina la primera vez que aparece elemento.

Código de ejemplo utilizando las operaciones más habituales

```
# Listas de ejemplo.
amigos=["Ariel Prado", "Felipe Marini"]
amigos2=["Ariel Prado", "Javier Acevedo"]

# Acceder a un elemento de la lista.
print(amigos[1])
Felipe Marini

# pop: Extraemos a Felipe (que está en la posición número 1) de la
lista. Ariel está en la posición 0.
amigos.pop(1)
print (amigos)
['Ariel Prado']

# append: Añadimos a Felipe al final de la lista
amigos.append("Felipe Marini")
print (amigos)
['Ariel Prado', 'Felipe Marini']

# del: Eliminamos el elemento de la primera posición de la lista (Ariel)
del(amigos[0])
print (amigos)
['Felipe Marini']

# insert: Añadimos a Ariel en la primera posición
amigos.insert(0, "Ariel Prado")
print (amigos)
['Ariel Prado', 'Felipe Marini']

# extend: Juntamos las listas de amigos. Ariel estará repetido.
amigos.extend(amigos2)
print (amigos)
['Ariel Prado', 'Felipe Marini', 'Ariel Prado', 'Javier Acevedo']

#remove: Borramos la primera vez que aparece Ariel Prado
amigos.remove("Ariel Prado")
print (amigos)
['Felipe Marini', 'Ariel Prado', 'Javier Acevedo']
```

Operaciones más habituales con diccionarios en Python

Es similar a las listas, con el matiz de que dado que los diccionarios no tienen orden, no tienen funciones en las que se tenga en cuenta la posición.

- `diccionario.get('key')`: Devuelve el valor que corresponde con la key introducida.
- `diccionario.pop('key')`: Devuelve el valor que corresponde con la key introducida, y luego borra la key y el valor.
- `diccionario.update({'key':'valor'})`: Inserta una determinada key o actualiza su valor si ya existiera.
- `"key" in diccionario`: Devuelve verdadero (True) o falso (False) si la key (no los valores) existe en el diccionario.
- `"definicion" in diccionario.values()`: Devuelve verdadero (True) o falso (False) si definición existe en el diccionario (como valor, no como key).

Código de ejemplo utilizando las operaciones más habituales

```
# Diccionario de ejemplo.
diccionario={'Amigo 1':'Ariel Prado', 'Amigo 2':'Felipe Marini', 'Amigo
3':'Javier Acevedo'}

# get(): Devuelve el valor que corresponde con la key introducida.
print(diccionario.get('Amigo 1'))
Ariel Prado

# pop(): Devuelve el valor que corresponde con la key introducida, y
luego borra la key y el valor.
print(diccionario.pop('Amigo 1'))
Ariel Prado
print(diccionario)
{'Amigo 3': 'Javier Acevedo', 'Amigo 2': 'Felipe Marini'}

# update(): Actualiza el valor de una determinada key o lo crea si no
existe.
diccionario.update({'Amigo 4':'Luis Rojo'})
diccionario.update({'Amigo 2':'Sebastián Bravo'})
print(diccionario)
{'Amigo 4': 'Luis Rojo', 'Amigo 2': 'Sebastián Bravo', 'Amigo 3':
'Javier Acevedo'}
```

```
# "key" in diccionario: devuelve verdadero (True) o falso (False) si la
key existe en el diccionario.
print ("Amigo 1" in diccionario)
True
print ("amigo 1" in diccionario) # Nótese comienzo en minúscula.
False
print ("Sebastián Bravo" in diccionario)
False # Da falso porque Sebastián Bravo no es una key

# "definición" in diccionario.values(): devuelve verdadero (True) o
falso (False) si la definición existe en el diccionario.
print ("Sebastián Bravo" in diccionario.values())
True

# del diccionario['key']: Elimina el valor (y el key) asociado a la key
indicada.
del diccionario['Amigo 2']
print(diccionario)
{'Amigo 4': 'Luis Rojo', 'Amigo 3': 'Javier Acevedo'}
```