# PROGRAMACIÓN CON PYTHON

(Interfaces punto de venta)

*Lenguajes de programación*

**Bautista Aguilar Fernando Angel 466658**
**Corona Ortiz Jerusalem 468460**
**Mejía Avecias Elizabeth 464307**
**Santiago Gonzales Yahir Fernando 464030**
**Serrano Manzano Gerardo Andrés 465780**

03/11/2022

# INTRODUCCIÓN

Python es un lenguaje de programación de propósito general muy potente y flexible, mientras que simple y fácil de aprender. Este lenguaje fue creado en los primeros años. 1990 por Guido van Rossum en los Países Bajos. Es software libre y es implementado en todas las plataformas y sistemas operativos comunes.

Adopta características de lenguajes anteriores e incluso hace compatible la solución Algunos. Por ejemplo, permite tres formas de imprimir el valor de una variable: desde el entorno interactivo escribiendo su nombre (como en Básico) usando el función print, con concatenación de elementos (al estilo del Write de Pascal) o con Patrón de formato (al estilo de C printf). Python está desarrollado bajo una licencia de código abierto o de código abierto aprobada por OSI para que pueda usarse y compartirse libremente, incluso con fines comerciales.

La Python Software Foundation (PSF) es una organización sin fines de lucro 501(c)(3). Quién posee los derechos de propiedad intelectual detrás del lenguaje de programación Python. Gestionamos las licencias de código abierto para Python versión 2.1 y luego.

El Python Package Index (PyPI) o en español significa Índice de Paquetes Python alberga miles de módulos de terceros para Python. Tanto la biblioteca estándar de Python como los módulos proporcionados por la comunidad permiten infinitas posibilidades. Entre esas posibilidades se encuentran las siguientes:

- Desarrollo web e Internet.
- Acceso a la base de datos.
- GUI’s de escritorio.
- Científico y numérico.
- Educación.
- Programación de red.
- Desarrollo de Software y Juegos.

Las características del lenguaje de programación Python se resumen a continuación:

- Es un lenguaje interpretado, no compilado, usa tipado dinámico, fuertemente tipado.
- Es multiplataforma, lo cual es ventajoso para hacer ejecutable su código fuente
- entre varios sistemas operativos.
- Es un lenguaje de programación multiparadigma, el cual soporta varios paradigma

- de programación como orientación a objetos, estructurada, programación
- imperativa y, en menor medida, programación funcional.
- En Python, el formato del código (p. ej., la indentación) es estructural.

## OBJETIVO

### Objetivo general.

Desarrollar una interfaz visual en Python que es permita hacer uso de diversos factores de este paradigma de programación.

### Objetivos específicos.

- Programar un sistema similar a un punto de venta sin bases de datos
- Generar las interfaces necesarias que permitan la manipulación de la información de entrada, dentro del sistema y de salida del sistema
- Crear interfaces claras, intuitivas y atractivas para los usuarios.

## DISCUSIÓN

Como equipo llegamos a la conclusión de que Python es un lenguaje de programación bastante versátil el cual puede tener distintos usos, como lo es la creación de interfaces, desarrollo web, inteligencia artificial, etc. Es una mezcla de varios lenguajes por lo que se pueden encontrar soluciones parecidas.

Sin duda es una buena herramienta que nos puede servir en el futuro hablando en el sentido del campo laboral ya que nos permite tener más cercanía con un entorno desde comandos a diferencia de otros que lo hacen un poco más gráfico.

# CUESTIONARIO

## 1. ¿Qué paradigmas de programación utiliza Python?

Python permite tres tipos de paradigmas;

- POO(Programación Orientada a Objetos)
- Programación orientada a procedimientos.
- Programación funcional.

## 2. ¿Cuál es la clasificación de los tipos de datos estándar en python?

Los tipos de datos básicos de Python son los booleanos, los numéricos (enteros, punto flotante y complejos) y las cadenas de caracteres.

Python también define otros tipos de datos, entre los que se encuentran:

- *Secuencias: Los tipos list, tuple y range*
- *Mapas: El tipo dict*
- *Conjuntos: El tipo set*
- *Iteradores*
- *Clases*
- *Instancias*
- *Excepciones*

## 3. ¿Cuáles son las palabras reservadas en Python, (escribe el listado y una breve descripción)?

- ***False*** – Valor booleano, resultado de operaciones de comparación u operaciones lógicas en Python
- ***None*** – Representa a un valor nulo
- ***True*** – Valor booleano, igual que false, resultado de operaciones de comparación u operaciones lógicas en Python
- ***__ peg_parser __*** – Llamado huevo de pascua, relacionado con el lanzamiento del nuevo analizador PEG no está definido aún.
- ***And –*** Operador lógico
- ***As –*** Se utiliza para crear un alias al importar un módulo.
- ***Assert –*** Se utiliza con fines de depuración

- ***Async –*** Proporcionada por la biblioteca ‘asyncio’ en Python. Se utiliza para escribir código concurrente en Python
- ***Await –*** Proporcionada por la biblioteca ‘asyncio’ en Python. Se utiliza para escribir código concurrente en Python
- ***Break –*** Se utiliza en el interior de los bucles for y while para alterar su comportamiento normal
- ***Class –*** Se usa para definir una nueva clase definida por el usuario
- ***Continue –*** Se utiliza en el interior de los bucles for y while para alterar su comportamiento normal
- ***Def –*** se usa para definir una función definida por el usuario
- ***Del –*** Para eliminar un objeto
- ***Elif –*** Se usa en declaraciones condicionales, igual ‘else’ e ‘if’
- ***Else –*** Se usa en declaraciones condicionales, igual ‘elif’ e ‘if’
- ***Except –*** Se usa para crear excepciones, qué hacer cuando ocurre una excepción, igual que ‘raise’ y ‘try’
- ***Finally –*** Su uso garantiza que el bloque de código dentro de él se ejecute incluso si hay una excepción no controlada
- ***For –*** Utilizado para hacer bucles. Generalmente lo usamos cuando sabemos la cantidad de veces que queremos que se ejecute ese bucle
- ***From –*** Para importar partes específicas de un módulo
- ***Global –*** Para declarar una variable global.
- ***If –*** Se usa en declaraciones condicionales, igual ‘else’ y ‘elif’
- Import – Para importar un módulo
- ***In –*** Para comprobar si un valor está presente en una lista, tupla, etc. Devuelve True si el valor está presente, de lo contrario devuelve False
- ***Is –*** Se usa para probar si las dos variables se refieren al mismo objeto. Devuelve True si los objetos son idénticos y False si no
- ***Lambda –*** Para crear una función anónima
- **Nonlocal –** Para declarar una variable no local
- ***Not –*** Operador lógico
- ***Or –*** Operador lógico
- ***Pass –*** Es una declaración nula en Python. No pasa nada cuando se ejecuta. Se utiliza como marcador de posición.
- ***Raise –*** Se usa para crear excepciones, qué hacer cuando ocurre una excepción, igual que ‘except y ‘try’

- ***Return*** **–** Se usa dentro de una función para salir y devolver un valor.
- ***Try*** **–** Se usa para crear excepciones, qué hacer cuando ocurre una excepción, igual que ‘raise’ y ‘except
- ***While*** **–** Se usa para realizar bucles.
- ***With*** **–** Se usa para simplificar el manejo de excepciones
- **Yield –** Se usa dentro de una función al igual que ‘return’, salvo que ‘yield’ devuelve un generador.

## 4. ¿Qué operadores de asignación se pueden utilizar en Python?

| OPERADOR | DESCRIPCIÓN |
|---|---|
| = | a = 5. El valor 5 es asignado a la variable a |
| += | a += 5 es equivalente a a = a + 5 |
| -= | a -= 5 es equivalente a a = a - 5 |
| *= | a *= 3 es equivalente a a = a * 3 |
| /= | a /= 3 es equivalente a a = a / 3 |
| %= | a %= 3 es equivalente a a = a % 3 |
| **= | a **= 3 es equivalente a a = a ** 3 |
| //= | a //= 3 es equivalente a a = a // 3 |
| &= | a &= 3 es equivalente a a = a & 3 |
| \|= | a \|= 3 es equivalente a a = a \| 3 |
| ^= | a ^= 3 es equivalente a a = a ^ 3 |
| >>= | a >>= 3 es equivalente a a = a >> 3 |
| <<= | a <<= 3 es equivalente a a = a << 3 |

## 5. ¿Cómo se realizan los comentarios de línea y multilínea en Python?

Comentarios en una sola línea: Para hacer un comentario de una sola línea solo basta con poner el signo de numeral al inicio de la línea. (Londoño, 2022)

# comentario.

```
1.  # Esto es un comentario en Python
2.  >>> funcion_1()
3.  >>> funcion_2() # Los comentarios también pueden estar después de un texto ejecutable
4.
5.  # Este es otro comentario en Python.
6.  # Puedes tener tantos comentarios de línea simple seguidos.
```

(Fernandez, 2020)

Comentarios multilínea: Para hacer un comentario multilínea se colocan tres comillas dobles al inicio y al final del bloque. (Londoño, 2022)

“ ” ”Comentarios
comentarios
comentarios” ” ”

```
1.  """Este es un comentario multilínea.
2.  Podemos escribir tantas líneas queramos a modo de documentación."""
3.  funcion_3()
4.  funcion_4()
5.
6.  '''También podemos hacer comentarios multilíneas con comillas simples.'''
```

(Fernandez, 2020)

## 6. ¿Qué es una lista en Python y describe los métodos que utiliza para la lista? ¿Qué es una tupla y cuáles son los métodos que se utilizan en Python para las tuplas?

Las listas en Python son un tipo contenedor, compuesto, que se usan para almacenar conjuntos de elementos relacionados del mismo tipo o de tipos distintos. Entre las más comunes se encuentran las siguientes:

***list.append(x)***

Agrega un ítem al final de la lista. Equivale a a[len(a):] = [x].

***list.extend(iterable)***

Extiende la lista agregándole todos los ítems del iterable. Equivale a a[len(a):] = iterable.

***list.insert(i, x)***

Inserta un ítem en una posición dada. El primer argumento es el índice del ítem delante del cual se insertará, por lo tanto a.insert(0, x) inserta al principio de la lista y a.insert(len(a), x) equivale a a.append(x).

***list.remove(x)***

Quita el primer ítem de la lista cuyo valor sea x. Lanza un ValueError si no existe tal ítem.

***list.pop([i])***

Quita el ítem en la posición dada de la lista y lo retorna. Si no se especifica un índice, a.pop() quita y retorna el último elemento de la lista. (Los corchetes que encierran a i en la firma del método denotan que el parámetro es opcional, no que deberías escribir corchetes en esa posición. Verás esta notación con frecuencia en la Referencia de la Biblioteca de Python.)

***list.clear()***

Elimina todos los elementos de la lista. Equivalente a del a[:].

***list.index(x[, start[, end]])***

Retorna el índice basado en cero del primer elemento cuyo valor sea igual a x. Lanza una excepción ValueError si no existe tal elemento.

Los argumentos opcionales start y end son interpretados como la notación de rebanadas y se usan para limitar la búsqueda a un segmento particular de la lista. El índice retornado se calcula de manera relativa al inicio de la secuencia completa en lugar de con respecto al argumento start.

***list.count(x)***

Retorna el número de veces que x aparece en la lista.

***list.sort(*, key=None, reverse=False)***
Ordena los elementos de la lista in situ (los argumentos pueden ser usados para personalizar el orden de la lista, ver sorted() para su explicación).

***list.reverse()***
Invierte los elementos de la lista in situ.

***list.copy()***
Retorna una copia superficial de la lista. Equivalente a a[:].

En cambio en Python, una tupla es un conjunto ordenado e inmutable de elementos del mismo o diferente tipo. Las tuplas se representan escribiendo los elementos entre paréntesis y separados por comas. Existen los siguientes métodos para las tuplas:

El método ***count()*** cuenta el número de veces que el objeto pasado como parámetro se ha encontrado en la lista.
El método ***index()*** busca el objeto que se le pasa como parámetro y devuelve el índice en el que se ha encontrado. En el caso de no encontrarse, se devuelve un ValueError. El método index() también acepta un segundo parámetro opcional, que indica a partir de qué índice empezar a buscar el objeto.

## 7. ¿Qué son los conjuntos en Python y cuales son los métodos que se utilizan para los conjuntos?

Un conjunto es una colección no ordenada de objetos únicos. Python provee este tipo de datos «por defecto» al igual que otras colecciones más convencionales como las listas, tuplas y diccionarios.

Los conjuntos son ampliamente utilizados en lógica y matemática, y desde el lenguaje podemos sacar provecho de sus propiedades para crear código más eficiente y legible en menos tiempo. Entre los tipos de métodos en los conjuntos tenemos los siguientes:

- ***add()***

  Añade un ítem a un conjunto, si ya existe no lo añade.

- ***discard()***

  Borra un ítem de un conjunto.

- ***copy()***

  Devuelva una copia de un conjunto, ya que éstos como la mayoría de colecciones se almacenan por referencia.

- ***clear()***

  Borra todos los ítems de un conjunto.

- ***union()***

  Une un conjunto a otro y devuelve el resultado en un nuevo conjunto.

- ***update()***

  Une un conjunto a otro en el propio conjunto.

- ***difference()***

  Encuentra los elementos no comunes entre dos conjuntos.

- ***difference_update()***

  Guarda en el conjunto los elementos no comunes entre dos conjuntos.

- ***intersection()***

  Devuelve un conjunto con los elementos comunes en dos conjuntos.

- ***intersection_update()***

  Guarda en el conjunto los elementos comunes entre dos conjuntos.

- ***symmetric_difference()***

  Devuelve los elementos simétricamente diferentes entre dos conjuntos, es decir, todos los elementos que no concuerdan entre los dos conjuntos.

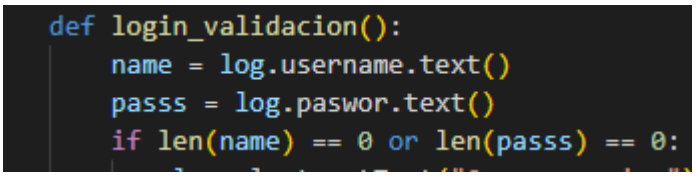
## 8. ¿Qué estructuras de control utilizaste en el desarrollo de la práctica?

Se utilizó la estructura de control “if ” principalmente para que se llevarán a cabo los diferentes eventos dentro del programa que es una estructura condicionar los diferentes casos o caminos que toma el programa según se reciban entradas generar salidas y se tenga respuesta a los diferentes casos.

## 9. Si utilizaste alguna de las funciones de Python, describe el procedimiento realizado para utilizarla

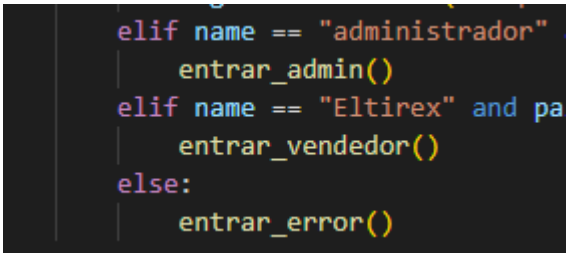
Tkinter es la forma de facto en Python para crear interfaces gráficas de usuario (GUI) y se incluye en todas las distribuciones estándar de Python. De hecho, es el único marco integrado en la biblioteca estándar de Python.

Hablando de funciones que nosotros implementamos empezábamos desde su creación y bien para crear funciones comenzábamos declarándose con la palabra “**def**” seguido del nombre de la función y después paréntesis y para finalizar sentencia con “**:**” como se muestra en la siguiente imagen.




Como se puede ver en la imagen se puede ver la estructura de una función en python en la que se declara su contenido con su respectivo identado(sangrado).

Después de esto para que se pueda definir la función dentro del segmento de código donde se requiere este fragmento con el nombre seguido de parentesis “**( )**” como se muestra en el siguiente ejemplo:




# REFERENCIAS BIBLIOGRÁFICAS

1. Covatec (2018), Programación en Python-nivel básico, disponible en: https://entrenamiento-python-basico.readthedocs.io/es/latest/index.html

2. López Ostenero Fernando, García Serrano Ana María (2014), Teoría de los Lenguajes de Programación, Editorial Universitaria Ramón Areces, ISBN: 9788499611396.

3. Tipos de datos básicos de Python - Cuáles son y características. (2022, 16 enero). J2LOGO. https://j2logo.com/python/tutorial/tipos-de-datos-basicos-de-python/

4. Peña, M. J. (2022, 26 agosto). Palabras reservadas en Python y su significado | EIP. Másteres Online No 1 Empleabilidad. https://eiposgrados.com/blog-python/palabras-reservadas-python/

5. Fernandez, R. (2020, 27 noviembre). Cómo hacer comentarios en Python. ▷ Cursos de Programación de 0 a Experto © Garantizados. https://unipython.com/como-hacer-comentarios-en-python/

6. Bustamante, S. J. (2021, 1 enero). Operadores Básicos en Python con ejemplos. freeCodeCamp.org. https://www.freecodecamp.org/espanol/news/operadores-basicos-en-python-con-ejemplos/

7. 5. Estructuras de datos — documentación de Python - 3.11.0. (s. f.). Recuperado 27 de octubre de 2022, de https://docs.python.org/es/3/tutorial/datastructures.html

8. Tuplas en Python. (s. f.). El Libro De Python. Recuperado 27 de octubre de 2022, de https://ellibrodepython.com/tuplas-python

9. Londoño, P. (2022, 20 julio). *Cómo hacer comentarios en Python (con ejemplos)*. https://blog.hubspot.es/website/hacer-comentarios-python