Sistemas de Gestión Empresarial

UF2 PAC3 - Sistemas ERP-CRM. Explotación y adecuación.

Aníbal Santos Gómez

1. **Define las características principales de Python.**

Python es un lenguaje de programación nacido en los años 90, cuyo objetivo es poder programar los servidores web de forma limpia y multiplataforma. Consigue la flexibilidad de los lenguajes script y la velocidad de un lenguaje ya compilado.

Sus principales características son:

- Es un lenguaje simple, por lo que es muy fácil iniciarse en este lenguaje.

- Es de propósito general.

- Es Open Source, puede funcionar gracias a esto en diversas plataformas.

- Es un lenguaje orientado a objetos, se construye sobre objetos que combinan datos y funcionalidades.

- Es de alto nivel.

- Es incrustrable, se puede insertar lenguaje Python dentro de un programa C/C++.

- Contiene una gran cantidad de librerías, tipos de datos y funciones incorporadas en el propio lenguaje, que ayudan a realizar muchas tareas comunes sin tener que programarlas desde cero.

- La sintaxis es muy clara, y visual, debido a la indexación.

2. **Características de las variables bajo Python.**

Las variables en Python tienen las siguientes caracterśiticas:

- No necesitan ser definidas para su utilización.

- Existen variables locales, que solo tienen alcance en el bloque que se inicialicen.

- Existen variables globales, que tienen un valor a lo largo del programa para las funciones y clases del mismo.

- En python las variables son etiquetas que permiten hacer referencia a los datos.

- Para cada dato Python crea un objeto y cada objeto tiene un identificador único, un tipo de datos y un valor.

- Python distingue entre objetos inmutables, que no se pueden modificar, y objetos mutables, que pueden ser modificados.

3. Describe los tipos numéricos básicos.

Los tipos de datos numéricos en python almacenan valores numéricos. Son tipos de datos inmutables, lo que significa que cuando cambia el valor del tipo de dato numérico el resultado es un objeto asignado nuevamente.

Python soporta cuatro tipos distintos numéricos:

- **int**, es un entero con signo, son números enteros positivos o negativos sin punto decimal.

- **long**, son números enteros de tamaño ilimitado. Escritos como enteros y seguidos de una maýsucula o minúscula.

- **float**, representan números reales y se escriben con un punto decimal dividiendo la parte entera y fraccional. Pueden estar en notación científica con E.

- **complex**, son de la forma a+bJ, donde a y b son flotantes y J, representa la raíz cuadrada de -1. a es la parte real del número y b es la parte imaginaria. No son demasiado utilizados en Python.

4. Diferencia entre las listas y las tuplas.

Podemos comparar listas y tuplas en otros lenguajes como arrays, pero poseen varias diferencias.

Ambas son un conjunto ordenado de valores, donde este último puede ser cualquier objeto, un número, una cadena, una función, una clase, una instancia, etc.

La diferencia es que las listas presentan una serie de funciones adicionales que permiten un amplio manejo de los valores que contienen.

Las listas son dinámicas mientras que las tuplas son estáticas.

Las listas pueden ser alteradas, pero las tuplas no. Las tuplas pueden ser utilizadas como clave en un diccionario pero las listas no. Una tupla consume menos espacio que una lista.

Una **lista** es una estructura de datos que contiene una colección o secuencia de datos. Los datos deben ir separados con coma y todo el conjunto entre corchetes. Es mutable porque permite el acceso a los elementos, agregación de nuevos elementos o supresión.

Una **tupla** permite tener agrupados un conjunto inmutable de elementos, es decir, en una tupla no es posible agregar ni eliminar elementos. Las tuplas se declaran separando los elementos por comas y estos pueden ir entre paréntesis.

5. **¿Qué bucle C no existe en Python?**

Una de las estructuras de control de ejecución que no está presente en Python es el ciclo **do-while**, suele darse que cuando requerimos utilizar do-while en Python para solicitar al usuario un dato de entrada que debe ser validado, la opción pertinente para sustituir a do-while sea e una cadena solicitada al usuario, r debería ser verdadero si contiene e, o falso en caso contrario, si r es falso, imprimimos error, después repetimos lo anterior mientras r sea falso, y por último imprimimos un mensaje de éxito.

Además de do-while, tampoco contamos con **switch-case**, esta estructura nos permite seleccionar por medio de una expresión, el siguiente bloque de instrucciones a ejecutar de entre varios tipos posibles. La forma de sustitución en Python es utilizar la secuencia de instrucciones if-elif-else. elif es una contracción de else e if.

6. **¿Cómo se gestionan los errores?**

En primer lugar tenemos **errores de sintaxis**, que son errores de interpretación. El intérprete repite la línea y muestra una flecha que apunta a lugar donde encuentra el error.

Respecto a las **excepciones**, son errores detectados durante la ejecución. Las excepciones vienen de distintos tipos, y el tipo se imprime como parte del mensaje, tenemos ZeroDivisionError, NameError y TypeError.

Es posible escribir programas que controlan determinadas excepciones. Para ello utilizamos la declaración try que funciona de la siguiente manera:

- Primero se ejecuta el bloque try.

- Si no hay excepción, el bloque except se salta y termina la ejecución de la declaración try.

- Si ocurre la excepción durante la ejecución del bloque try, el resto del bloque se salta. Si su tipo coincide con la excepción nombrada después de la palabra reservada except, se ejecuta el bloque except y la ejecución continúa después de la declaración try.

- Si ocurre una excepción que no coincide con la excepción nombrada en el bloque except, esta se pasa a las declaraciones try, si no se encuentra nada que la maneje, es una excepción no controlada, y la ejecución se frena con un mensaje.

7. **Tipo de herencia para crear un módulo bajo OpenERP.**

Dentro de la herencia orientada a OpenERP, es posible diferenciar entre tres tipos, **_name, _inherit** e **_inherits**.

- La **herencia por extensión**, permite añadir atributos a un objeto existente, se utiliza en el caso en el que las propiedades _name e _inherit sean iguales.

- La **herencia por prototipo**, va a crear un objeto nuevo que va a heredar todos los componentes del padre, aunque es independiente. Se crea en el caso de que _name tenga un valor no registrado e

_inherit tenga un valor registrado. Por lo que se añaden al objeto nuevo todos los campos definidos en el heredado.

- La **herencia por delegación**, se usa cuando es necesario crear un objeto a partir de otros ya existentes. Estableceremos un nombre de objeto no registrado en _name y crearemos el diccionario correspondiente con todos los objetos existentes que vayan a heredar, con _inherits. El nuevo objeto tendrá todos los campos de todos los objetos, más todos los que se le añadan.

8. Tipos de vistas existentes en un módulo OpenERP.

Dentro de OpenERP tenemos las siguientes vistas:

- **Vistas formularios.** Son aquellas que se utilizan para cumplimentar los campos necesarios para un registro. Los campos distribuidos en las vistas formularios siempre son de forma determinada, se distribuyen en la pantalla de izquierda a derecha y de arriba a abajo, y cada pantalla se divide en cuatro columnas.

- **Vistas árbol.** Se utilizan cuando se trabaja en modo de lista y en la pantalla de búsqueda. Este tipo de vista es más simple que el de formulario y por lo tanto tienen menos opciones.

- **Vistas gráfico.** Se caracterizan por ser totalmente dinámicas, se puede hacer click en cualquier parte del gráfico, arrastrar y soltar el objeto en otra ubicación.

- **Vistas proceso.** Los nodos y transiciones de los objetos pueden ser visualizados en este tipo de vistas. Puede obtenerse además información acerca de las transiciones entre los nodos. Podemos obtener información de la descripción de la transición, una lista de funciones que puede llevar a cabo la transición, y las acciones disponibles para el estado.

- **Vistas Gantt.** Todos los objetos del sistema pueden ser visualizados a través de esta vista, esto es muy útil en planificación de recursos humanos y materiales.

9. **¿Qué tipo de vista son graph y search?**

- **Graph**. es un tipo de vista para que los distintos formularios puedan mostrar un gráfico que se forma a partir de determinados datos.

- **Search**. es un tipo de vista de búsqueda complementario a la vista en árbol, que añade un panel de búsqueda y un filtrado. Se utilizan en los listados de datos.