Integrantes:

Sergio Esteban Pacheco

Armand Daniel ceballos

Ficha:

2749613

fecha:

11/10/2023

1. ¿Qué es un lenguaje de programación?

RST:

Un lenguaje de programación es un idioma que usamos para comunicarnos con una computadora, celular y/o tablet para poder decirle que queremos hacer.

1. ¿Qué diferencia hay entre un intérprete y un compilador?

RST:

La diferencia es que el intérprete procesa el proyecto de software durante su tiempo de ejecución y el compilador lo que hace es transformar el proyecto a un código ejecutable.

1. ¿Qué es un programa?

RST:

Un programa es el conjunto de pasos lógicos escritos en un lenguaje de programación para decirle a una computadora que hay que hacer en una serie de pasos para realizar una tarea específica.

1. Tipos de errores en programación

RST:

* Nombres arbitrarios
* falta de originalidad
* quedarse con lo teórico
* no pedir ayuda
* abarcar muchos temas al mismo tiempo
* error de sintaxis
* proceso no válido
* lógicos resultados no incorrectos
* lógicos tipo bucle infinito

1. ¿Qué es Debugging?

RST:

Debugging es un sistema para detectar errores y poder solucionar los errores que se encuentran en el código de cualquier software.

1. ¿Qué es python?

RST:

Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web.

1. Explique variables, tipos de datos, operaciones y expresiones básicas de python.

RST:

* Una variante es un espacio designado en la memoria de una computadora para almacenar datos. Se puede ver como un contenedor con un nombre que se refiere a un valor o un conjunto de valores.
* **tipos de datos en programación:**

Enteros (int):

representan números enteros sin ningún decimal, pueden ser positivos o negativos

.

Numeros de punto flotante (float):

Permite representar números con decimales.

cadenas de texto (string):

representa secuencias de caracteres como letras, números y símbolos.

Booleanos (bool):

puede ser entre solo dos tipos de valores, verdadero (true) o falso (false), sirven para la lógica del programa

.

Listas y arreglos (list, array):

permiten almacenar una colección de valores bajo un mismo nombre.

Diccionarios (dict):

asocian en pares de clave-valor, es decir acceder a un valor por su clave.

tuplas (tuple):

Es similar a la lista, pero esta es inmutable, es decir que no se puede modificar después de su creación.

conjuntos (set):

Almacenan colecciones de elementos únicos, no se permite hacer un duplicado.

ninguno (none):

representa la ausencia de un valor o la falta de una definición.

* **Tipos de operaciones en programación:**

Operaciones Aritméticas:

- Suma (+): Combina dos o más valores numéricos.

- Resta (-): Resta un valor de otro.

- Multiplicación (\*): Realiza una multiplicación entre dos o más valores.

- División (/): Divide un valor por otro.

- Módulo (%): Devuelve el residuo de la división entre dos valores.

- Potencia (\*\*): Eleva un número a una potencia.

Operaciones de Comparación:

- Igualdad (==): Compara si dos valores son iguales.

- Desigualdad (!= o <>): Compara si dos valores no son iguales.

- Mayor que (>): Comprueba si un valor es mayor que otro.

- Menor que (<): Comprueba si un valor es menor que otro.

- Mayor o igual que (>=): Comprueba si un valor es mayor o igual que otro.

- Menor o igual que (<=): Comprueba si un valor es menor o igual que otro.

Operaciones Lógicas:

- Y (and): Retorna verdadero si ambas condiciones son verdaderas.

- O (or): Retorna verdadero si al menos una de las condiciones es verdadera.

- No (not): Niega una condición, convierte verdadero en falso y viceversa.

Operaciones de Asignación:

- Asignación (=): Asigna un valor a una variable.

- Operadores de asignación compuesta (+=, -=, \*=, /=, %=, etc.): Realizan una operación y luego asignan el resultado a la variable.

Operaciones de Concatenación (para cadenas):

- Concatenación (+): Combina dos o más cadenas de texto.

Operaciones de Pertenencia:

- In (en): Verifica si un valor está presente en una secuencia (como una lista o una cadena).

Operaciones de Identidad:

- is (es): Su función es comprobar si dos variables apuntan al mismo objeto en la memoria.

* **EXPRESIONES BÁSICAS DE PYTHON**

6. Expresiones

Este capítulo explica el significado de los elementos de expresiones en Python.

Notas de Sintaxis: En este y los siguientes capítulos será usada notación BNF extendida para describir sintaxis, no análisis léxico. Cuando (una alternativa de) una regla de sintaxis tiene la forma,

name ::= other name. y no han sido dadas semánticas, las semánticas de esta forma de name son las mismas que para otro name.

1. Conversiones aritméticas.

* Si cualquiera de los argumentos es un número complejo, el otro es convertido a complejo;
* de otra forma, si cualquier de los argumentos es un número de punto flotante, el otro es convertido a punto flotante;
* de otra forma, ambos deben ser enteros y no se necesita conversión.

2. Átomos.

Los átomos son los elementos más básicos de las expresiones. Los átomos más simples son identificadores o literales. Las formas encerradas en paréntesis, corchetes o llaves son también sintácticamente categorizadas como átomos. La sintaxis para átomos es:

**atom** ::=  [identifier](https://docs.python.org/es/3/reference/lexical_analysis.html#grammar-token-python-grammar-identifier) | [literal](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-literal) | [enclosure](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-enclosure) **enclosure** ::=  [parenth\_form](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-parenth_form) | [list\_display](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-list_display) | [dict\_display](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-dict_display) | [set\_display](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-set_display) | [generator\_expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-generator_expression) | [yield\_atom](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-yield_atom)

3. Primarios

Los primarios representan las operaciones más fuertemente ligadas al lenguaje. Su sintaxis es:

**primary** ::=  [atom](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-atom) | [attributeref](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-attributeref) | [subscription](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-subscription) | [slicing](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-slicing) | [call](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-call)

4 Referencias de atributos

Una referencia de atributo es un primario seguido de un punto y un nombre:

**attributeref** ::=  [primary](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-primary) "." [identifier](https://docs.python.org/es/3/reference/lexical_analysis.html#grammar-token-python-grammar-identifier)

5. El operador de potencia

El operador de potencia se vincula más estrechamente que los operadores unarios a su izquierda; se vincula con menos fuerza que los operadores unarios a su derecha. La sintaxis es:

**power** ::= ([await\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-await_expr) | [primary](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-primary)) ["\*\*" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr)]

6. Aritmética binaria y operaciones bit a bit

Toda la aritmética binaria y las operaciones bit a bit tienen la misma prioridad:

**u\_expr** ::=  [power](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-power) | "-" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr) | "+" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr) | "~" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr)

7. Operaciones aritméticas binarias

Las operaciones aritméticas binarias tienen los niveles convencionales de prioridad. Tenga en cuenta que algunas de esas operaciones también aplican a ciertos tipos no numéricos. Aparte del operador de potencia, hay sólo dos niveles, uno para operadores multiplicativos y uno para aditivos:

**m\_expr** ::=  [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr) | [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) "\*" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr) | [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) "@" [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) |

[m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) "//" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr) | [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) "/" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr) |

[m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) "%" [u\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-u_expr)

**a\_expr** ::=  [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) | [a\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-a_expr) "+" [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr) | [a\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-a_expr) "-" [m\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-m_expr)

8. Operaciones de desplazamiento

Las operaciones de desplazamiento tienen menos prioridad que las operaciones aritméticas:

**shift\_expr** ::=  [a\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-a_expr) | [shift\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-shift_expr) ("<<" | ">>") [a\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-a_expr)

9. Operaciones bit a bit binarias

Cada una de las tres operaciones de bits binarias tienen diferente nivel de prioridad:

**and\_expr** ::=  [shift\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-shift_expr) | [and\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-and_expr) "&" [shift\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-shift_expr)

**xor\_expr** ::=  [and\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-and_expr) | [xor\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-xor_expr) "^" [and\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-and_expr)

**or\_expr** ::=  [xor\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-xor_expr) | [or\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_expr) "|" [xor\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-xor_expr)

10. Comparaciones

A diferencia de C, todas las operaciones de comparación en Python tienen la misma prioridad, la cual es menor que la de cualquier operación aritmética, de desplazamiento o bit a bit. También, a diferencia de C, expresiones como a < b < c tienen la interpretación convencional en matemáticas:

**comparison** ::=  [or\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_expr) ([comp\_operator](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-comp_operator) [or\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_expr))\*

**comp\_operator** ::= "<" | ">" | "==" | ">=" | "<=" | "!="

| "is" ["not"] | ["not"] "in"

11. Operaciones booleanas

**or\_test** ::=  [and\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-and_test) | [or\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_test) "or" [and\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-and_test)

**and\_test** ::=  [not\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-not_test) | [and\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-and_test) "and" [not\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-not_test)

**not\_test** ::=  [comparison](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-comparison) | "not" [not\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-not_test)

12. Expresiones de asignación

**assignment\_expression** ::= [[identifier](https://docs.python.org/es/3/reference/lexical_analysis.html#grammar-token-python-grammar-identifier) ":="] [expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-expression)

13. Expresiones condicionales

**conditional\_expression** ::=  [or\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_test) ["if" [or\_test](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_test) "else" [expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-expression)]

**expression** ::=  [conditional\_expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-conditional_expression) | [lambda\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-lambda_expr)

14. Lambdas

**lambda\_expr** ::= "lambda" [[parameter\_list](https://docs.python.org/es/3/reference/compound_stmts.html#grammar-token-python-grammar-parameter_list)] ":" [expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-expression)

15. Listas de expresiones

**expression\_list** ::=  [expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-expression) ("," [expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-expression))\* [","]

**starred\_list** ::=  [starred\_item](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-starred_item) ("," [starred\_item](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-starred_item))\* [","]

**starred\_expression** ::=  [expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-expression) | ([starred\_item](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-starred_item) ",")\* [[starred\_item](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-starred_item)]

**starred\_item** ::=  [assignment\_expression](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-assignment_expression) | "\*" [or\_expr](https://docs.python.org/es/3/reference/expressions.html#grammar-token-python-grammar-or_expr)

16. Orden de evaluación

Python evalúa las expresiones de izquierda a derecha. Note que mientras se evalúa una asignación, la parte derecha es evaluada antes que la parte izquierda.

En las siguientes líneas, las expresiones serán evaluadas en el orden aritmético de sus sufijos:

expr1, expr2, expr3, expr4(expr1, expr2, expr3, expr4){expr1: expr2, expr3: expr4}expr1 + expr2 \* (expr3 - expr4)expr1(expr2, expr3, \*expr4, \*\*expr5)expr3, expr4 = expr1, expr2

bibliografia

<https://docs.python.org/es/3/reference/expressions.html#arithmetic-conversions>

* **OPERACIONES QUE SE PUEDEN HACER EN PYTHON**

| **Clases de operadores de Python** | **Explicación** | **Operandos** | **Resultado** | **Resumen de operadores** |
| --- | --- | --- | --- | --- |
| **Operadores aritméticos** | **Unen dos cifras y crean una nueva** | **Cifras** | **Cifra** | **+, -, \*, /, //, %, \*\*, @** |
| **Operadores de comparación** | **Comparan dos expresiones entre sí** | **Expresiones** | **Booleano** | **<, >, ==, !=, <=, >=** |
| **Operadores lógicos** | **Vinculan expresiones en contexto booleano** | **Expresiones** | **Última expresión evaluada/booleano** | **and, or, not** |
| **Operadores bit a bit** | **Manipulan números enteros como secuencias binarias** | **Cifras** | **Cifra** | **<<, >>, &, |, ^, ~** |
| **Operadores de asignación** | **Asignan un nombre a un valor** | **Lvalue, Rvalue** | **– / expresión evaluada** | **=, :=, +=, -=, \*=, etc.** |
| **Operadores de identidad** | **Determinan si dos nombres se refieren al mismo objeto** | **Objetos** | **Booleano** | **is, is not** |
| **Operadores condicionales** | **Devuelve uno de dos valores en función de una condición** | **Expresión, condición, alternativa** | **Expresión/alternativa** | **... if ... else ...** |
| **Operadores de conjuntos** | **Unen dos conjuntos/enlazan conjuntos** | **Conjuntos** | **Conjunto/booleano** | **&, |, ^, -, <, >, <=, >=** |
| **Operadores de membresía** | **Prueban si un iterable contiene un objeto determinado** | **Objeto, iterable** | **booleano** | **in, not in** |
| **Operador de concatenación** | **Concatena secuencias** | **Strings / listas / registro** | **String / lista / registro** | **+** |
| **Operadores índex y slicing** | **Entre uno o más elementos iterables** | **Iterable, índex / Slice** | **String / Lista / registro** | **[], [::]** |

**bibliografía**

[**https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/python-operators/**](https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/python-operators/)