**KEYWORDS**

|  |  |
| --- | --- |
| **Keywords** | **Description** |
| class | class is used to define a new user-defined class in Python. |
| return | return statement is used inside a function to exit it and return a value. |
| break, skip | break and skip are used inside for and while loops to alter their normal behavior. |
| if,then,otherwise | If else structure of our language. Programmer shoud use if and then in one statement. |
| for | for is used for looping. Generally we use for when we know the number of times we want to loop. |
| until | until is used as while loop in our program. |
| do...until | Executes a statement block once, and then repeats execution of the loop until a condition expression evaluates to false. |
| for\_each | foreach loop. |
| Start | main class |
| New | Creates a new object. |
| Null | An object that represents "no object". |
| super | Refers to the base object of the current object. This can be used in two contexts. |
| try, catch | except, try are used with exceptions. |
| finally | finally is used with try…except block to close up resources or file streams. Using finally ensures that the block of code inside it gets executed even if there is an unhandled exception. |
| current | Refers to the current object. |
| Final | Declares that a class cannot be extended or that a method or property cannot be overridden. |
| private | Declares that a class member has private visibility. |
| public | Declares that a class, or member has public visibility. |
| Protected | Visibility within a package |
| Static | Declares that a class member belongs to a class rather than to instances of the class. |
| abstract | Used as an interface for base class |
| override | Used for late binding |
| virtual | Used for multiple inheritance. (Diamond problem) |
| switch, case, default |  |
| Void | Prevents an expression from returning a value. |
| throw, throws | Use for exceptions. |

|  |  |  |  |
| --- | --- | --- | --- |
| PRIMITIVE | REFERENCE | SIZE MODIFIERS | SIGN MODIFIERS |
| int | String | short | signed |
| char | File | long | unsigned |
| float | Class |  |  |
| double |  |  |  |
| byte: 0b for binary, 0x for hexadecimal, 0 for octal | Array |  |  |

**DATA TYPES**

# OPERATORS

# Arithmetic Operators:

|  |  |
| --- | --- |
| OPERATORS | DISCRIPTION |
| + | Addition − Adds values on either side of the operator. |
| - | Subtraction − Subtracts right hand operand from left hand operand. |
| \* | Multiplication − Multiplies values on either side of the operator. |
| / | Division − Divides left hand operand by right hand operand. |
| % | Modulus − Divides left hand operand by right hand operand and returns remainder. |
| ++ | Increment |
| -- | Decrement |
| \*\* | Exponent − Performs exponential (power) calculation on operators. |

## Assignment Operators:

|  |  |
| --- | --- |
| OPERATORS | DISCRIPTION |
| **:** | Simple assignment operator, assigns values from right side operands to left side operand. |

## Comparison Operators:

|  |  |
| --- | --- |
| OPERATORS | DISCRIPTION |
| Equals | Checks if the value of two operands are equal or not, if yes then condition becomes true. |
| not\_equals | Checks if the value of two operands are equal or not, if values are not equal then condition becomes true. |
| less\_than | Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true. |
| greater\_than | Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true. |
| greater\_than\_equal | Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true. |
| less\_than\_equal | Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true. |

## Logical Operators:

|  |  |
| --- | --- |
| OPERATORS | DISCRIPTION |
| And | Called Logical AND operator. If both the operands are non zero, then the condition becomes true. |
| Or | Called Logical OR Operator. If any of the two operands are non zero, then the condition becomes true. |
| Not | Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true, then Logical NOT operator will make false. |

## Bitwise Operators:

|  |  |
| --- | --- |
| OPERATORS | DISCRIPTION |
| Andb | Binary AND Operator copies a bit to the result if it exists in both operands. |
| Orb | Binary OR Operator copies a bit if it exists in either operand |
| Notb | Binary Ones Complement Operator is unary and has the effect of 'flipping' bits. |
| Xor | Binary XOR Operator copies the bit if it is set in one operand but not both. |
| left\_shift | Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand. |
| right\_shift | Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand. |

**Punctuators:**

|  |  |
| --- | --- |
| OPERATORS | DISCRIPTION |
| **.** | Property accessor. |
| **,** | Argument delimiter |
| **:** | Part of conditional operator and used in switch-case statements. |
| **\r** | Statement terminator |
| **<** | Parameter open for Container Class |
| **>** | Parameter close for Container Class |
| ( | Function argument delimiter and precedence control. |
| ) | Function argument delimiter and precedence control. |
| [ | Array index delimiter. |
| ] | Array index delimiter. |
| { | Start code block. |
| } | End code block |
| **|** | Used to separate exceptions in try..catch. |

**OOP CONCEPTS & FEATURES**

**-**multiple inheritance is allowed

-polymorphism should be used explicitly

-container classes:

Set: cannot contain duplicate elements

List: ordered collection, can contain duplicates

Hashtable: key/value pair

-object class