# Python variables

|  |  |
| --- | --- |
| **Example** | **Type** |
| Last\_Name = Delgado | String |
| Score = 9.83 | Float |
| Approved = True | Boolean |
| Number\_List = 3 | Int |

# Strings

|  |  |
| --- | --- |
| **Example** | **Description** |
| Course\_Name = “Ultimate python” | We use “ ” to print character strings |
| Course\_Description = “ “ “ This course teaches python skills to get a job as a programmer ” ” ” | We use “ “ “ ” ” ” to print to many character strings |
| Course\_Name = “Ultimate python”  len(Course\_Name) | We use len(Course\_Name) to print the length of the character string we assign to the in a variable.  print(len(Course\_Name)) 🡪 15  Length is 15 characters |
| Course\_Name = “Ultimate python”  print(Course\_Name[0]) | We use [0], [1], [2], [3], …[n]. inside print to print the specific character of the character string  print(Course\_Name[0]) 🡪 U  print(Course\_Name[1]) 🡪 I  print(Course\_Name[2]) 🡪 T  print(Course\_Name[9]) 🡪 P |
| Course\_Name = “Ultimate python”  print(Course\_Name[0:8]) | We use [n:n] inside print to cut and print the characters strings  print(Course\_Name[0:8]) 🡪 Ultimate  print(Course\_Name[9:]) 🡪 Python  print(Course\_Name[:8]) 🡪 Ultimate  print(Course\_Name[:])🡪 Ultimate Python |
| Name = Ozzy  Last\_Name = Aceves  Full\_Name = f ”{ Name } { Last\_Name } ” | When we want to format strings, we use  Full\_Name = f ”{ Name } { Last\_Name } to format strings  When we have:  Full\_Name = f ”{ Name } { Last\_Name }  print(Full\_Name) 🡪 Ozzy Aceves  When we have:  Full\_Name = f ”{ Name[0] } { 2 + 5 }  print(Full\_Name) 🡪 O 7 |
| Animal = “Happy Monkey”  print(Animal.upper()) | We use .upper to convert all characters to uppercase within the string  print(Animal.upper()) 🡪 HAPPY MONKEY |
| Animal = “Happy Monkey”  print(Animal.lower()) | We use .lower to convert all characters to lowercase within the string  print(Animal.lower()) 🡪 happy monkey |
| Animal = “Happy Monkey”  print(Animal.capitalize()) | We use .capitalize to convert the first character in uppercase within the string  print(Animal.capitalize()) 🡪 Happy monkey |
| Animal = “Happy Monkey”  print(Animal.title()) | We use .title to convert the firsts characters in uppercase within the string  print(Animal.title()) 🡪 Happy Monkey |
| Animal = “Happy Monkey”  print(Animal.strip()) | We use .strip to remove spaces at the beginning and at the end of the string  print(Animal.strip()) 🡪 Happy Monkey |
| Animal = “Happy Monkey”  print(Animal.strip().strip()) | We use .capitalize().strip() (or other combination) to convert the first character in uppercase within the string and remove spaces at the beginning and at the end of the string  print(Animal.capitalize().strip()) 🡪 Happy monkey |
| Animal = “Happy Monkey”  print(Animal.lstrip()) | We use .lstrip to remove the spaces to the left of the string  print(Animal.lstrip()) 🡪 Happy Monkey |
| Animal = “Happy Monkey”  print(Animal.rstrip()) | We use .rstrip to remove the spaces to the right of the string  print(Animal.rstrip())🡪 Happy Monkey |
| Animal = “Happy Monkey”  print(Animal.find(“pp”)) | We use .find(“pp”) to search characters within the strings and return the indice  print(Animal.find(“pp”)) 🡪 3  print(Animal.find(“zy”)) 🡪 -1 this case not found the characters and return -1 |
| Animal = “Happy Monkey”  print(Animal.replace(“pp”, “k”)) | We use .replace(“pp”, “k”) to replace characters within the strings  print(Animal.replace(“pp”, “k”)) 🡪 Haki Monkey |
| Animal = “Happy Monkey”  print( “pp” in Animal) | We use print( “pp” in Animal) to search characters within the string and return a boolean (True or False)  print( “pp” in Animal) 🡪 True  print( “zy” in Animal) 🡪 False |
| Animal = “Happy Monkey”  print( “pp” not in Animal) | We use print( “pp” not in Animal) to search characters within the string and return a boolean (True or False)  print( “pp” not in Animal) 🡪 False  print( “zy” not in Animal) 🡪 True |

# Sequences and escapes

|  |  |
| --- | --- |
| Course = “Ultimate \”Python”\”  print(Course) | We use \“ \” when we want to write the character “ ” within strings and another “ ”  print(Course) 🡪 Ultimate “Python” |
| Course = ’Ultimate ”Python”’  print(Course) | We use ‘ ’ when we want to write the character “ ” within strings and another “ ”  print(Course) 🡪 Ultimate “Python” |
| #Comment | We use the symbol # to comment the code |
| Course = “Ultimate \nPython”  print(Course) | We use \n when we want to write the following characters in the following line  print(Course) 🡪 Ultimate  Python |
| Course = “Ultimate \\Python”  print(Course) | We use \\ when we want to write the character \ within strings  print(Course) 🡪 Ultimate \Python |

# Numbers

|  |  |
| --- | --- |
| Number\_1 = input( “Enter the number:” ) | We use input() when we want to store a data in a variable |
| Number\_Integer = 2 | This number is integer |
| Number\_Float = 4.2 | This number is float |
| Number\_Imaginary = 2 + 2j | This number is a imaginary |
| Number\_Integer = 2  Number\_Integer += 5 | The expression Number\_Integer = Number\_Integer + 5 can be written in another form  Number\_Integer += 5 |
| Number\_Integer\_2 = 4  Number\_Integer\_2 -= 5 | The expression Number\_Integer = Number\_Integer - 5 can be written in another form  Number\_Integer -= -1 |
| Number\_Integer\_3 = 6  Number\_Integer\_3 \*= 5 | The expression Number\_Integer = Number\_Integer \* 5 can be written in another form  Number\_Integer \*= 30 |
| Number\_Integer\_4 = 8  Number\_Integer\_4 += 5 | The expression Number\_Integer = Number\_Integer / 5 can be written in another form  Number\_Integer /= 1.6 |
| print(1 // 3) | We use // in the operation to print the result without decimals  print(1 // 3) 🡪 0 |
| print(1 % 3) | We use % in the operation to print the residue of the division  print(1 % 3) 🡪 1 |
| print(1 \*\* 3) | We use \*\* in the operation as an operator raised to power  print(1 \*\* 3) 🡪 1 |
| print(round(1.2))  print(round(1.5))  print(round(1.7)) | We use round(n) in the operation to round off the result  print(round(1.2)) 🡪 1  print(round(1.5)) 🡪 2  print(round(1.7))🡪 2 |
| print(abs(-77))  print(abs(55)) | We use abs(n) in the operation to print the absolute value  print(abs(-77)) 🡪 77  print(abs(55)) 🡪 55 |
| Import math  print(math.ceil(1.1)) | We use math.ceil(n) in the operation to round up the number  print(math.ceil(1.1)) 🡪 2 |
| Import math  print(math.floor (1.999)) | We use math.floor(n) in the operation to round down the number  print(math.floor(1.999)) 🡪 1 |
| Import math  print(math.isnan(23)) | We use math.isnan(n) to print whether or not the value is a number  print(math.isnan(23)) 🡪 False |
| Import math  print(math.pow(10, 3)) | We use math.pow(n) in the operation to raise a number to a certain power  print(math.pow(10, 3)) 🡪 1000.0 |
| Import math  print(math.sqrt(9)) | We use math.sqrt(n) in the operation to get the square root of the number  print(math.sqrt(9)) 🡪 3 |

# Type conversion

|  |  |
| --- | --- |
| Number\_1 = input( “Enter the number:” )  Number\_1 = int(Number\_1)  print(int(Number\_1)) | We use Number\_1 = int(Number\_1) to convert a string into an integer  We use print(int(Number\_1)) to convert and print a string into an integer |
| Number\_1 = input( “” )  Number\_1 = float(Number\_1)  print(float(Number\_1)) | We use Number\_1 = float(Number\_1) to convert a string into a float  We use print(float(Number\_1)) to convert and print a string into a float |
| Number\_2 = input()  Number\_2 = str(Number\_2)  print(str(Number\_2)) | We use Number\_2 = str(Number\_2) convert a data into a string  We use print(str(Number\_2)) to convert and print a data into a string |
| print(bool(“”))  print(bool(“ “))  print(bool(“0”))  print(bool(None))  print(bool(0)) | We use print(bool()) to convert and print a data into bool  print(bool(“”)) 🡪 False  print(bool(“ “)) 🡪 True  print(bool(“0”)) 🡪 True  print(bool(None)) 🡪 False  print(bool(0)) 🡪 False |

# Logic comparators

|  |  |
| --- | --- |
| print(1 < 2) | We use < to indicate that the data or value is less than another one  print(1 < 2) 🡪 True  print(3 < 2) 🡪 False |
| print(1 > 2) | We use > to indicate that the data or value is greater than another one  print(1 > 2) 🡪 False  print(3 > 2) 🡪 True |
| print(1 <= 2) | We use <= to indicate that the data or value is less than or equal to another  print(1 <= 2) 🡪 True  print(2 <= 2) 🡪 True  print(3 <= 2) 🡪 False |
| print(1 >= 2) | We use >= to indicate that the data or value is greater than or equal to another  print(1 >= 2) 🡪 False  print(2 >= 2) 🡪 True  print(3 >= 2) 🡪 True |
| print(1 == 2) | We use == to indicate that the data or value is equal to another one  print(1 == 2) 🡪 False  print(2 == 2) 🡪 True  print(2 == “2”) 🡪 False |
| print(1 != 2) | We use != to indicate that the data or value is different from another one  print(1 != 2) 🡪 True  print(2 != 2) 🡪 False  print(2 != “2”) 🡪 True |

# If, elif, else

|  |  |
| --- | --- |
| 1. Age = 33 2. if Age > 53: 3. print(“You can watch the movie at a discount”) 4. elif Age > 21: 5. print(“You can watch the movie”) 6. else: 7. print(“You can’t enter”) 8. print(“You must go elsewhere”) 9. print(“End”) | We use if to evaluate a statement, and elif to add another statement and so on and so forth, and we add else in case none of the above sentences are met |