1. What is a variable? How to create variable? What are the rules for declaring variables?

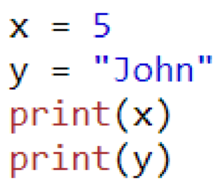
Variables

Variables are containers for storing data values.

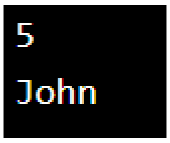
Creating variables

Python has no command for declaring a variable.

A variable is created the moment you first assign a value to it.



```python
x = 5
y = "John"
print(x)
print(y)
```



```
5
John
```

Variable Names

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume). Rules for Python variables:

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age and AGE are three different variables)
- A variable name cannot be any of the Python keywords.

2. What are strings? What are the functions associated with the string?

Strings in python are surrounded by either single quotation marks, or double quotation marks.

String Length

To get the length of a string, use the len() function.

Example

The len() function returns the length of a string:

```
a = "Hello, World!"
print(len(a))
```

```
13
```

Check String

To check if a certain phrase or character is present in a string, we can use the keyword in.

Use it in an if statement:

Example

Print only if "free" is present:

```
txt = "The best things in life are free!"
if "free" in txt:
  print("Yes, 'free' is present.")
```

```
Yes, 'free' is present.
```

Check if NOT

To check if a certain phrase or character is NOT present in a string, we can use the keyword not in.

Example

print only if "expensive" is NOT present:

```
txt = "The best things in life are free!"
if "expensive" not in txt:
  print("No, 'expensive' is NOT present.")
```

```
No, 'expensive' is NOT present.
```

3. **What is list? What are the functions associated with the list?**
   Lists are used to store multiple items in a single variable.

List Length

To determine how many items a list has, use the len() function:

Example

Print the number of items in the list:

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

```
3
```

Append Items

To add an item to the end of the list, use the append() method:

Example

Using the append() method to append an item:

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
print(thislist)
```

```
['apple', 'banana', 'cherry', 'orange']
```

Insert Items

To insert a list item at a specified index, use the insert() method.

The insert() method inserts an item at the specified index:

Example

Insert an item as the second position:

```
thislist = ["apple", "banana", "cherry"]
thislist.insert(1, "orange")
print(thislist)
```

```
['apple', 'orange', 'banana', 'cherry']
```

Remove Specified Item

The remove() method removes the specified item.

Example

Remove "apple":

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("apple")
print(thislist)
```

```
['banana', 'cherry']
```

Remove Specified Index

The pop() method removes the specified index.

Example

Remove the second item:

```
thislist = ["apple", "banana", "cherry"]
thislist.pop(1)
print(thislist)
```

4. What is tuple? What are the functions associated with the tuple?

Tuples are used to store multiple items in a single variable.

Tuple Length

To determine how many items a tuple has, use the len() function:

Example

Print the number of items in the tuple:

```
thistuple = ("apple", "banana", "cherry")
print(len(thistuple))
```

```
3
```

Add Items

Since tuples are immutable, they do not have a built-in append() method, but there are other ways to add items to a tuple.

1. **Convert into a list**: Just like the workaround for changing a tuple, you can convert it into a list, add your item(s), and convert it back into a tuple.

Example

Convert the tuple into a list, add "orange", and convert it back into a tuple:

```
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.append("orange")
thistuple = tuple(y)
```

```
('apple', 'banana', 'cherry', 'orange')
```

Remove Items

Note: You cannot remove items in a tuple.

Tuples are **unchangeable**, so you cannot remove items from it, but you can use the same workaround as we used for changing and adding tuple items:

Example

Convert the tuple into a list, remove "apple", and convert it back into a tuple:

```
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.remove("apple")
thistuple = tuple(y)
```

```
('banana', 'cherry')
```

Or you can delete the tuple completely:

Example

The del keyword can delete the tuple completely:

```
thistuple = ("apple", "banana", "cherry")
del thistuple
print(thistuple) #this will raise an error because the tuple no longer exists
```

```
Traceback (most recent call last):
  File "demo_tuple_del.py", line 3, in <module>
    print(thistuple) #this will raise an error because the tuple no longer exists
NameError: name 'thistuple' is not defined
```

5. Explain slicing and string format method.

Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

Example

Get the characters from position 2 to position 5 (not included):

```
b = "Hello, World!"
print(b[2:5])
```

```
llo
```

Note: The first character has index 0.

Slice From the Start

By leaving out the start index, the range will start at the first character:

Example

Get the characters from the start to position 5 (not included):

```
b = "Hello, World!"
print(b[:5])
```

```
Hello
```

Slice To the End

By leaving out the end index, the range will go to the end:

Example

Get the characters from position 2, and all the way to the end:

```
b = "Hello, World!"
print(b[2:])
```

```
llo, World!
```

String Format

We can combine strings and numbers by using the format() method!

The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

```
quantity = 3
itemno = 567
price = 49.95
myorder = "I want {} pieces of item {} for {} dollars."
print(myorder.format(quantity, itemno, price))
```

```
I want 3 pieces of item 567 for 49.95 dollars.
```

6. Explain data type conversion in python.

You can convert from one type to another with the int(), float(), and complex() methods:

```
x = 1    # int
y = 2.8  # float
z = 1j   # complex

#convert from int to float:
a = float(x)

#convert from float to int:
b = int(y)

#convert from int to complex:
c = complex(x)

print(a)
print(b)
print(c)

print(type(a))
print(type(b))
print(type(c))
```

```
1.0
2
(1+0j)
<class 'float'>
<class 'int'>
<class 'complex'>
```

7. **Explain**
    a. **Identity operators**
    b. **Membership operators**

Python Identity Operators

Identity operators are used to compare the objects, not if they are equal, but if they are actually the same object, with the same memory location:

Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

Python Membership Operators

Membership operators are used to test if a sequence is presented in an object:

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

8. **Explain**
    a. **Break statement**
    b. **Continue statement**

The break Statement

With the break statement we can stop the loop before it has looped through all the items:

Example

Exit the loop when x is "cherry":

```
fruits = ["apple", "banana", "cherry","orange"]
for x in fruits:
  print(x)
  if x == "cherry":
    break
```

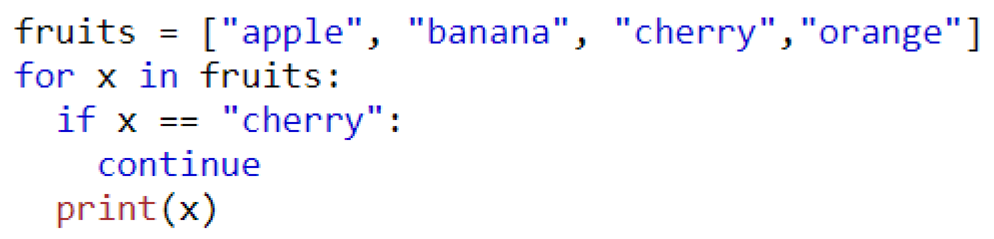
```
apple
banana
cherry
```

The continue Statement

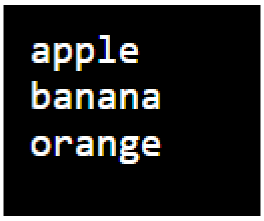
With the continue statement we can stop the current iteration of the loop, and continue with the next:

Example

Do not print cherry:



```
fruits = ["apple", "banana", "cherry","orange"]
for x in fruits:
  if x == "cherry":
    continue
  print(x)
```



```
apple
banana
orange
```

9. Explain the working of if condition along with elif and else with example?

An "if statement" is written by using the if keyword.

Python relies on indentation (whitespace at the beginning of a line) to define scope in the code. Other programming languages often use curly-brackets for this purpose.

If statement, without indentation will raise an error

The elif keyword is Python's way of saying "if the previous conditions were not true, then try this condition".

The else keyword catches anything which isn't caught by the preceding conditions.

Example

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

```
a is greater than b
```

In this example a is greater than b, so the first condition is not true, also the elif condition is not true, so we go to the else condition and print to screen that "a is greater than b".

10.Explain for loop? How to use for loop to loop through a string?

For Loops

A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).

With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.

Example

Print each fruit in a fruit list:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

```
apple
banana
cherry
```

The for loop does not require an indexing variable to set beforehand.

Looping Through a String

Even strings are iterable objects, they contain a sequence of characters:

Example

Loop through the letters in the word "apple":

```
for x in "apple":
  print(x)
```

```
a
p
p
l
e
```

11.Explain the concept of random number.

Random Number

Python does not have a random() function to make a random number, but Python has a built-in module called random that can be used to make random numbers:

Example

Import the random module, and display a random number between 1 and 9:

```
import random

print(random.randrange(1, 10))
```

```
8
```

12.Explain while loop with an example.

The while Loop

With the while loop we can execute a set of statements as long as a condition is true.

Example

Print i as long as i is less than 6:

```python
i = 1
while i < 6:
  print(i)
  i += 1
```

```
1
2
3
4
5
```

Note: remember to increment i, or else the loop will continue forever.

The while loop requires relevant variables to be ready, in this example we need to define an indexing variable, i, which we set to 1.

Multiple choice questions.

1. What is the data type of the variable 'x' in Python, if x = 5?
   A) **Integer**
   B) String
   C) Float
   D) Boolean
2. What data type would you use to store a sequence of characters in Python?
   A) Integer
   B) Float
   C) **String**
   D) Boolean
3. What will be the data type of the result of 10 * 2.5 in Python?
   A) Integer
   B) **Float**
   C) String
   D) Boolean
4. What is the data type of the result of the expression "Hello" + "World" in Python?
   A) Integer
   B) Float
   C) **String**
   D) Boolean
5. What will be the data type of the result of 10 > 5 in Python?
   - A) Integer
   - B) Float
   - C) String
   - D) **Boolean**
6. What function in Python is used to determine the length of a string?
   A) **len**()

B) length()
C) str_len()
D) strlen()

7. Which of the following methods in Python is used to convert a string to uppercase?
A) to_upper()
B) uppercase()
C) **upper**()
D) toUpperCase()
8. Which method in Python is used to remove leading and trailing whitespace characters from a string?
A) trim()
B) **strip**()
C) clean()
D) remove()
9. What function in Python is used to replace occurrences of a substring within a string with another substring?
A) **replace**()
B) swap()
C) substitute()
D) modify()
10. What function in Python is used to format strings with placeholders?
A) **format**()
B) interpolate()
C) insert()
D) merge()
11. What is the result of 5 + 3 in Python?
A) **8**
B) 15
C) 53
D) Error
12. Which operator in Python is used for string concatenation?
A) &
B) +
C) *
D) %
13. What is the result of the expression 10 > 5 in Python?
A) **True**
B) False
C) 1
D) Error
14. Which looping statement in Python executes a block of code repeatedly as long as a condition is true?
A) for loop
B) **while loop**
C) do-while loop
D) until loop
15. In Python, how can you exit from a loop prematurely?
A) **Using the break statement**
B) Using the stop statement
C) Using the exit statement

D) Using the end statement

16. What does the 'continue' statement do in Python?
    A) Exits the loop
    B) **Skips the current iteration and proceeds to the next iteration**
    C) Stops the program
    D) Restarts the loop from the beginning
17. Which loop in Python is typically used when you know the exact number of iterations?
    A) while loop
    B) **for loop**
    C) do-while loop
    D) until loop
18. What keyword is used for making decisions in Python?
    A) decide
    B) choose
    C) **if**
    D) switch
19. Which of the following is the correct syntax for an if statement in Python?
    A) if condition { }
    B) **if condition: (with indentation for block)** C) if condition then { }
    D) if condition; { }
20. What does the 'elif' keyword signify in Python?
    A) **It is short for "else if" and is used to check multiple conditions**.
    B) It indicates the start of a loop.
    C) It means "else" in Python.
    D) It indicates a comment.
21. In Python, what is the purpose of the 'else' statement?
    A) It is used for error handling.
    B) It is used for input/output operations.
    C) It is used to execute a block of code if the condition is true.
    D) **It is used to execute a block of code if the condition is false.**
22. How do you define a tuple in Python?
    A) Using square brackets [ ]
    B) **Using parentheses ( )**
    C) Using curly braces { }
    D) Using angle brackets < >
23. How do you define a list in Python?
    A) **Using square brackets [ ]**
    B) Using parentheses ( )
    C) Using curly braces { }
    D) Using angle brackets < >
24. How do you define an empty dictionary in Python?
    A) empty_dict = []
    B) **empty_dict = {}**
    C) empty_dict = ()
    D) empty_dict = None
25. What is the main purpose of using dictionaries in Python?
    A) To store elements in a sorted order
    B) To store elements with duplicate values
    C) **To store elements as key-value pairs**

D) To store elements in a stack