Extra class 1

Henri Vandendriessche
henri.vandendriessche@ens.fr

2023-12-19

MCQ data variable

- Remove incorrect characters in the name of the variable

```
1_of_my-variable! = "test"
```

MCQ boolean 1

- What is the value of the following statement ?

```
print(5>1)
```

MCQ boolean 1

- What is the value of the following statement ?

```
print(5>1)
```

```
## True
```

MCQ boolean 2

- What is the value of the following statement ?

```
print(5 == 1)
```

MCQ boolean 2

- What is the value of the following statement ?

```
print(5 == 1)
```

```
## False
```

MCQ boolean 3

- What is the value of the following statement ?

```
print(5 < 1)
```

MCQ boolean 3

- What is the value of the following statement ?

```
print(5 < 1)
```

```
## False
```

MCQ boolean 4

- What is the value of the following statement ?

```
print(bool("abc"))
```

MCQ boolean 4

- What is the value of the following statement ?

```
print(bool("abc"))
```

```
## True
```

MCQ boolean 5

- What is the value of the following statement ?

```
print(bool(0))
```

MCQ boolean 5

- What is the value of the following statement ?

```
print(bool(0))
```

```
## False
```

MCQ boolean 6

- What is the value of the following statement ?

```
print(bool(1))
```

MCQ boolean 6

- What is the value of the following statement ?

```
print(bool(1))
```

```
## True
```

MCQ data type

- What will be printed from the following examples ?

```
x = 5
print(type(x))
```

```
x = "Hello World"
print(type(x))
```

```
x = 20.5
print(type(x))
```

```
x = ["cat", "dog", "horse"]
print(type(x))
```

```
x = ("cat", "dog", "horse")
print(type(x))
```

```
x = {"name" : "John", "lastname" : "Doe", "age" : 33}
print(type(x))
```

```
x = True
print(type(x))
```

MCQ Strings 1

- Get the first character of the string txt

```
txt = "Hello World"
x = ?
```

MCQ Strings 1

- Get the first character of the string txt

```
txt = "Hello World"
x = txt[0]
print(x)
```

```
## H
```

MCQ Strings 2

- Get the character from index 2 to 4 (llo)

```
txt = "Hello World"
x =
```

MCQ Strings 2

- Get the character from index 2 to 4 (llo)

```
txt = "Hello World"
x = txt[2:5]
print(x)
```

```
## llo
```

MCQ Strings 3

- Return the string without any whitespace at the beginning

```
txt = " Hello World"
x =
```

MCQ Strings 3

- Return the string without any whitespace at the beginning

```
txt = " Hello World"
print(txt)
```

```
##  Hello World
```

```
x = txt[1:]
print(x)
```

```
## Hello World
```

```
y = txt.strip()
print(y)
```

```
## Hello World
```

MCQ Strings 4

- Convert the text in upper case

```
txt = "Hello World"
x =
```

MCQ Strings 4

- Convert the text in upper case

```
txt = "Hello World"
x = txt.upper()
print(x)
```

```
## HELLO WORLD
```

MCQ Strings 5

- Convert the text in lower case

```
txt = "Hello World"
x =
```

MCQ Strings 5

- Convert the text in lower case

```
txt = "Hello World"
x = txt.lower()
print(x)
```

```
## hello world
```

MCQ Strings 6

- Replace H by J

```
txt = "Hello World"
x =
```

MCQ Strings 6

- Replace H by J

```
txt = "Hello World"
x = txt.replace("H", "J")
print(x)
```

```
## Jello World
```

MCQ operator 1

- Use the correct membership operator to check if “cat” is present in the animal object.

```
animal = ["cat", "dog"]
if "cat"
```

MCQ operator 1

- Use the correct membership operator to check if “cat” is present in the animal object.

```
animal = ["cat", "dog"]
if "cat" in animal:
  print("Yes, cat is a animal!")
```

```
## Yes, cat is a animal!
```

MCQ operator 2

- Use the correct comparison operator to check if 5 is not equal to 1.

```
if
```

MCQ operator 2

- Use the correct comparison operator to check if 5 is not equal to 1.

```
if 5 != 10:
  print("5 and 10 is not equal")
```

```
## 5 and 10 is not equal
```

MCQ operator 3

- Use the correct logical operator to check if at least one of two statements is True.

```
if 5 == 10 ?? 4 == 4:
  print("At least one of the statements is true")
```

MCQ operator 3

- Use the correct logical operator to check if at least one of two statements is True.

```
if 5 == 10 or 4 == 4:
  print("At least one of the statements is true")
```

```
## At least one of the statements is true
```

```
if (5 == 10) | (4 == 4):
  print("At least one of the statements is true")
```

```
## At least one of the statements is true
```

MCQ List 1

- Print the third item in the fruits list.

```
animal = ["cat", "dog","horse"]
print()
```

MCQ List 1

- Use the correct logical operator to check if at least one of two statements is True.

```
animal = ["cat", "dog","horse"]
print(animal[2])
```

```
## horse
```

MCQ List 2

- Change the value from “cat” to “lion”, in the fruits list.

```
animal = ["cat", "dog","horse"]
```

MCQ List 2

- Change the value from “cat” to “lion”, in the fruits list.

```
animal = ["cat", "dog","horse"]
animal[0] = "lion"
print(animal)
```

```
## ['lion', 'dog', 'horse']
```

MCQ List 3

- Add cow to the animal list

```
animal = ["cat", "dog","horse"]
```

MCQ List 3

- Add cow to the animal list

```
animal = ["cat", "dog","horse"]
animal.append("cow")
print(animal)
```

```
## ['cat', 'dog', 'horse', 'cow']
```

MCQ List 4

- remove dog to the animal list

```
animal = ["cat", "dog","horse"]
```

MCQ List 4

- remove dog to the animal list

```
animal = ["cat", "dog","horse"]
animal.remove("dog")
print(animal)
```

```
## ['cat', 'horse']
```

MCQ List 5

- Use negative indexing to print the last item in the list.

```
animal = ["cat", "dog","horse"]
```

MCQ List 5

- Use negative indexing to print the last item in the list.

```
animal = ["cat", "dog","horse"]
print(animal[-1])
```

```
## horse
```

MCQ List 6

- Use the correct syntax to print the number of items in the list.

```
animal = ["cat", "dog","horse"]
```

MCQ List 6

- Use the correct syntax to print the number of items in the list.

```
animal = ["cat", "dog","horse"]
print(len(animal))
```

```
## 3
```

MCQ List 7

- Use a range of indexes to print the third, fourth, and fifth item in the list.

```
animal = ["cat", "dog","horse"]
```

MCQ List 7

- Use a range of indexes to print the third, fourth, and fifth item in the list.

```
animal = ["cat", "dog","horse"]
print(animal[2:5])
```

```
## ['horse']
```

MCQ Dictionaries 1

- Use the get method to print the value of the “model” key of the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print()
```

MCQ Dictionaries 1

- Use the get method to print the value of the “model” key of the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(car.get("model"))
```

```
## Mustang
```

MCQ Dictionaries 2

Change the “year” value from 1964 to 2023.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

MCQ Dictionaries 2

Change the “year” value from 1964 to 2023.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
car["year"]=2023
print(car)
```

```
## {'brand': 'Ford', 'model': 'Mustang', 'year': 2023}
```

MCQ Dictionaries 3

Add the key/value pair “color” : “red” to the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

MCQ Dictionaries 3

Add the key/value pair “color” : “red” to the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
car["color"]="red"
print(car)
```

```
## {'brand': 'Ford', 'model': 'Mustang', 'year': 1964, 'color': 'red'}
```

MCQ Dictionaries 4

Use the pop method to remove “model” from the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

MCQ Dictionaries 4

Use the pop method to remove “model” from the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
car.pop("model")
```

```
## 'Mustang'
```

```
print(car)
```

```
## {'brand': 'Ford', 'year': 1964}
```

MCQ Dictionaries 5

Use the clear method to empty the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

MCQ Dictionaries 5

Use the clear method to empty the car dictionary.

```
car =   {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
car.clear()
print(car)
```

```
## {}
```

MCQ If 1

- Print “Yes” if a is equal to b, otherwise print “No”.

```
a = 50
b = 10
```

MCQ If 1

- Print “Yes” if a is equal to b, otherwise print “No”.

```
a = 50
b = 10
if a == b:
  print("Yes")
else:
  print("No")
```

```
## No
```

MCQ If 3

Print “Hello” if a is equal to b, and c is equal to d.

```
a = 50
b = 10
c = 30
d = 30
```

MCQ If 3

Print “Hello” if a is equal to b, and c is equal to d.

```
a = 50
b = 10
c = 30
d = 30
if a == b and c==d:
  print("Hello")
```

MCQ For 1

Loop through the items in the animal list.

```
animal = ["cat", "dog","horse"]
```

MCQ For 1

Loop through the items in the animal list.

```
animal = ["cat", "dog","horse"]
for idx in animal:
  print(idx)
```

```
## cat
## dog
## horse
```

MCQ For 2

In the loop, when the item value is “dog”, jump directly to the next item.

```
animal = ["cat", "dog","horse"]
for idx in animal:
  if idx == "dog":
    ?????
  print(idx)
```

MCQ For 2

In the loop, when the item value is “dog”, jump directly to the next item.

```
animal = ["cat", "dog","horse"]
for idx in animal:
  if idx == "dog":
    continue
  print(idx)
```

```
## cat
## horse
```

MCQ For 3

Use the range function to loop through a code set 5 times.

```
for x in ???? :
  print(x)
```

MCQ For 3

Use the range function to loop through a code set 5 times.

```
for x in range(5) :
  print(x)
```

```
## 0
## 1
## 2
## 3
## 4
```

MCQ While 1

- Print i as long as i is less than 5.

```
i = 1
```

MCQ While 1

- Print i as long as i is less than 5.

```
i = 1
while i < 5:
  print(i)
  i += 1
```

```
## 1
## 2
## 3
## 4
```

MCQ While 2

- Exit the loop if i $= 3$

```
i = 1
while i < 5:
  i += 1
  if i == 3:
    ?????
  print(i)
```

MCQ While 2

- Exit the loop if i = 3

```
i = 1
while i < 5:
  i += 1
  if i == 3:
    break
  print(i)
```

```
## 2
```

MCQ Function 1

- Create a function named my_function.

```
??????:
  print("Hello from a function")
```

MCQ Function 1

- Create a function named my_function.

```
def my_function():
  print("Hello from a function")
```

MCQ Function 2

- Let the function return the x parameter + 5.

```
var = 5

def my_function(x):
  ???????

my_function(var)
```

MCQ Function 2

- Let the function return the x parameter + 5.

```
var = 5

def my_function(x):
  return x+5

my_function(var)
```

```
## 10
```

MCQ Function 3

- If you do not know the number of arguments that will be passed into your function, there is a prefix you can add in the function definition, which prefix?

```
def my_function(???kids):
  print("The youngest child is " + kids[2])
```

MCQ Function 3

- If you do not know the number of arguments that will be passed into your function, there is a prefix you can add in the function definition, which prefix?

```
def my_function(*kids):
  print("The youngest child is " + kids[2])
```

MCQ Function 4

- If you do not know the number of keyword arguments that will be passed into your function, there is a prefix you can add in the function definition, which prefix?

```
def my_function(???kids):
  print("The youngest child is " + kids[2])
```

MCQ Function 4

- If you do not know the number of keyword arguments that will be passed into your function, there is a prefix you can add in the function definition, which prefix?

```
def my_function(**kid):
  print("His last name is " + kid["lname"])
```

MCQ Module 1

- What is the correct syntax to import a module named “mymodule”?

```
??? my_module
```

MCQ Module 1

- What is the correct syntax to import a module named “mymodule”?

```
import my_module
```

MCQ Module 2

- If you want to refer to a module by using a different name, you can create an alias.

What is the correct syntax for creating an alias for a module?

```
import mymodule ?? mx
```

MCQ Module 2

- If you want to refer to a module by using a different name, you can create an alias.

What is the correct syntax for creating an alias for a module?

```
import mymodule as mx
```

MCQ Module 3

- What is the correct syntax of printing all variables and function names of the "random" module?

```
import random
print(????)
```

MCQ Module 3

- What is the correct syntax of printing all variables and function names of the “mymodule” module?

```
import random
print(dir(random))
```

```
## ['BPF', 'LOG4', 'NV_MAGICCONST', 'RECIP_BPF', 'Random', 'SG_MAGICCONST', 'SystemRandom
     ', 'TWOPI', '_Sequence', '_Set', '__all__', '__builtins__', '__cached__', '__doc__',
     '__file__', '__loader__', '__name__', '__package__', '__spec__', '_accumulate', '
     _acos', '_bisect', '_ceil', '_cos', '_e', '_exp', '_floor', '_inst', '_log', '_os', '
     _pi', '_random', '_repeat', '_sha512', '_sin', '_sqrt', '_test', '_test_generator', '
     _urandom', '_warn', 'betavariate', 'choice', 'choices', 'expovariate', 'gammavariate
     ', 'gauss', 'getrandbits', 'getstate', 'lognormvariate', 'normalvariate', '
     paretovariate', 'randbytes', 'randint', 'random', 'randrange', 'sample', 'seed', '
     setstate', 'shuffle', 'triangular', 'uniform', 'vonmisesvariate', 'weibullvariate']
```

MCQ Module 4

- What is the correct syntax of importing only the randint function of the “random” module?

```
??? random ??? randint
```

MCQ Module 4

- What is the correct syntax of importing only the randint function of the “random” module?

```
from random import randint
```