PREPARAÇÃO PARA O TE

Todas as perguntas dos LE’s e mais

- Todas as perguntas estão misturados e não há nenhuma ordem
- Algumas perguntas são do manual e não dos LE’s

Consider that somebody types at the Python interpreter prompt:

```
1 + 2 + 3
```

What is the output?

Selecione uma opção de resposta:

- nothing
- `b=6 of type int`
- Runtime error
- `6`

Consider that somebody types at the Python interpreter prompt:

```
a_int = 1 + 2 + 3
```

What is the output?

Selecione uma opção de resposta:

- `6 of type int`
- Runtime error
- nothing
- `6`

Consider the following English sentence:

```
boy eats cat
```

The sentence is:

Selecione uma opção de resposta:

- Syntactically incorrect
- Syntactically correct
- Runtime incorrect
- Runtime correct

Consider the following English sentence:

```
The boy eats a cat.
```

The sentence is:

Selecione uma opção de resposta:

- Runtime incorrect
- Semantically incorrect
- Runtime correct
- Semantically correct

Given the following code fragment what is its Big-O running time?

```
cut = 0
for i in range(n):
        for j in range(n):
                cut += i * j
```

- O(n)
- O(n^2)
- O(n log n)
- O(n^3)

Consider the following Python statement:

```
a = 1 + 2
```

The statement is:

Selecione uma opção de resposta:

- Runtime correct
- Syntactically correct
- Runtime incorrect
- Syntactically incorrect

What command correctly generates the sequence 3, 6, 9?

Selecione uma opção de resposta:

- `range(9, 6, -3)`
- `range(3, 10, 3)`
- `range(3, 6, 9)`
- `range(2, 9, 3)`

The following code contains an *infinite* loop:

```
n = 10
answer = 1
while n > 0:
    answer = answer + n
    n = n + 1
print(answer)
```

Which is the best explanation for why the loop does not terminate?

Selecione uma opção de resposta:

- In the *while-loop* body, we must set `n` to `False` and this code does not do that
- `answer` starts at `1` and is incremented by `n` each time, so it will always be positive
- One cannot compare `n` to `0` in a *while-loop*; it must be compared with another variable
- `n` starts at `10` and is incremented by `1` each time through the loop, so it will always be positive

Assume you want to produce the following table:

```
0 0
0 1
1 1
```

What is correct Python code?

Selecione uma opção de resposta:

-
```
for i in range(3):
    for j in range(i, 2):
        print(i, j)
```
-
```
for i in range(3):
    for j in range(2):
        print(i, j)
```
-
```
for i in range(1, 2):
    for j in range(i, 2):
        print(j, i)
```
-
```
for i in range(2):
    for j in range(2):
        print(i, j)
```

Consider the following command submitted to the Python interpreter:

```
print(6 % 2)
```

What is the output?

Selecione uma opção de resposta:

- A runtime error
- `3 of type int`
- `3`
- `0`

Consider the following command submitted to the Python interpreter:

```
print(2 * (3 - 4))
```

What is the output?

Selecione uma opção de resposta:

- A runtime error
- `2`
- Nothing
- `-2`

Consider the following code submitted to the Python interpreter:

```
for number in [3, 2, 1, 0]:
    print("I have", number, "cookies and I'm going to eat one.")
```

How many lines does this code print?

Selecione uma opção de resposta:

- 4
- 3
- 5
- 0

Consider `x = 2`, `y = 3`, and `z = 4` and the following command submitted to the Python interpreter:

```
if x < y and x < z:
  print("a")
elif y < x and y < z:
  print("b")
else:
  print("c")
```

What is the output?

Selecione uma opção de resposta:

- c
- a
- b
- None, it gives a syntactic error

In order to execute a function, we need to make a function call.

Which of the following **must** be provided in the function call?

Selecione uma opção de resposta:

- Function name and a *Return* value
- Function name and a *Docstrings*
- Formal parameters, if needed
- Actual parameters, if needed

Consider the following command submitted to the Python interpreter:

```
type("Hello World!")
```

What is the output?

Selecione uma opção de resposta:

- `<class 'str'>`
- Nothing
- `Hello World!`
- A runtime error

Consider the expression:

```
5*3 > 10 and 4+6 == 10
```

Which of the following properly expresses the precedence of operators (using parentheses)?

Selecione uma opção de resposta:

- `((5*3) > 10) and ((4+6) == 10)`
- `((5*3) > (10 and (4+6))) == 10`
- `(5*(3 > 10)) and (4 + (6 == 10))`
- `(((5*3) > 10) and 4)+6) == 10`

Consider the following code submitted to the Python interpreter:

```
a_str = 2+2 == 2
print(a_str)
```

What is the output?

Selecione uma opção de resposta:

- `4`
- `False`
- `True`
- None, it gives a syntactic error

What is the difference between a tab ('`\t`') and a sequence of spaces?

Selecione uma opção de resposta:

- A tab will line up items in a second column, regardless of how many characters were in the first column, while spaces will not
- There is no difference
- A tab is wider than a sequence of spaces
- Tabs are used for creating tables which cannot be done using spaces

Which type of loop can be used to perform the following iteration: choose a positive integer at random and then print the numbers from 1 up to and including the selected integer?

Selecione uma opção de resposta:

- only a *while-loop*
- only a *for-loop*
- a *for-loop* or a *while-loop*
- None, it gives a *runtime* error

To define a function, we need to provide a *header* and a *body*.

What must be in the definition *header*?

Selecione uma opção de resposta:

- `def f_name:`
- `def f_name ( parameter, ... )`
- `def f_name ( parameter, ... ):`
- `def f_name ( [parameter, ...] ):`

When we create a local variable in the body of a Python function, can we use it outside the body of that function?

Selecione uma opção de resposta:

- Yes, but only after the function call
- No, we'll get an error
- Yes, the lifetime of variables span to the rest of the program
- Yes, local variables are global in Python

When we define a function, how many statements must be present in the *body*?

Selecione uma opção de resposta:

- Zero or more statements plus **return**
- Zero or more statements
- One or more statements
- A *docstring*, zero or more statements plus **return**

Void functions are not intended to return a value, as there's no `return` statement in their body.

Which of the following is true for Python 3.7 *void functions*?

Selecione uma opção de resposta:

- Without a return, there will be a *runtime* error
- The function returns the value `Void`
- The function returns the value `None`
- The function does not return any value

Consider the following command submitted to the Python interpreter:

```
True = "true"
```

What is the output?

Selecione uma opção de resposta:

- Nothing
- A syntactic error
- `<class 'bool'>`
- `True`

Can you rewrite any *for-loop* as a *while-loop* and vice-versa?

Selecione uma opção de resposta:

- It is possible to rewrite any *for-loop* as a *while-loop* but not vice-versa
- It is possible to rewrite any *while-loop* as a *for-loop* but not vice-versa
- It is not always possible
- Yes

Consider the following variable definitions:

```
s1 = "The quick brown"
s2 = "fox"
s3 = "jumps"
```

and the following command submitted to the Python interpreter:

```
print(s1 + s2 + s3)
```

What is the output?

Selecione uma opção de resposta:

- Nothing
- `The quick brownfoxjumps`
- A runtime error
- `The quick brown fox jumps`

What is the output of the following statements?

```
list(range(2, 0, -1))
```

Selecione uma opção de resposta:

- `range(2, 0, -1)`
- `[2, 0, -1]`
- `[2, 1]`
- `(2, 0, -1)`

What is the output of the following statements?

```
S1 = {1, 2, 3, 4}
S2 = {3, 4, 5, 6}
print(S1 | S2)
```

Selecione uma opção de resposta:

- None, there's a *runtime* error
- `{1, 2, 3, 4, 5, 6}`
- `{3, 4}`
- `{1, 2})`

What is the output of the following statements?

```
mydict = {"Cleese":80, "Idle":76, "Palin":77}
answer = mydict.get("Palin", 0) // mydict.get("Chapman", 1)
print(answer)
```

Selecione uma opção de resposta:

- `0`
- None, there's a *runtime* error (division is not a valid operation on dictionaries)
- None, there's a *runtime* error (there's no entry with "Chapman" as the key)
- `77`

What is the output of the following statements?

```
mydict = {"Cleese":80, "Idle":76, "Palin":77}
tydict = mydict
tydict["Cleese"] = 77
print(mydict["Cleese"])
```

Selecione uma opção de resposta:

- `80`
- `None`
- None, there's a *runtime* error (there are two different keys named "Cleese")
- `77`

What is the output of the following statements?

```
mydict = {"Cleese":80, "Idle":76, "Palin":77}
mydict["total"] = mydict["Idle"] + mydict["Palin"]
print(mydict["total"])
```

Selecione uma opção de resposta:

- None, there's a *runtime* error (there is no entry with "total" as the key in `mydict`)
- `0`
- `807677`
- `153`

What does the following function return?

```
def f(x):
    if x < -2 and x > 5:
        return x
```

Selecione uma opção de resposta:

- Always returns `None`
- Returns numbers `x` below -2 and over 5 and gives *runtime error* otherwise
- Returns numbers `x` below -2 and over 5 and returns `None` otherwise
- Always returns the actual parameter for `x`

Which of the following is unit testing?

- Test if a variable value is valid
- Test PEP8 style
- Test the code comments
- Test the output of a function given a certain input

What is the output of the following statements?

```
nums = [2, 4, 6]
nums[0] = "beer"
print(nums[0])
```

Selecione uma opção de resposta:

- None, there's a syntax error
- `"beer"`
- `2`
- None, there's a *runtime* error

What is the output of the following statements?

```
s1 = "Super"
s2 = "Bock"
(s1, s2) = (s2, s1)
print(s1 + " " + s2)
```

Selecione uma opção de resposta:

- None, it produces a *runtime* error
- `Bock Super`
- `('Bock', 'Super')`
- `('Super', 'Bock')`

What are pdb and Python Tutor?

- Unit testers
- Debuggers
- Variable inspectors
- Documentation generators

What is the output of the following statements?

```
ax = (23,)
print(type(ax))
```

Selecione uma opção de resposta:

- `<class 'list <int>'>`
- `<class 'int'>`
- `<class 'tuple'>`
- None, it produces a *runtime* error

What is the output of the following statements?

```
s = "Cleese"
t = "+"
print(s+t*3)
```

Selecione uma opção de resposta:

- None, it gives a syntax error
- `Cleese+++`
- `Cleese+Cleese+Cleese+`
- `Cleese***`

Which one is a suitable comment?

- i += 1 #incrementing i
- i += 1 #go to next element
- i += 1 #this line is very important
- i += 1 #adding 1 to i

What happens to statements after the `return` and in the same block of code?

Selecione uma opção de resposta:

- That is a syntactic error in Python
- There will be a *runtime* error
- They are interpreted if inside a boolean function
- They are *dead code* that is never interpreted

What is the output of the following statement?

```
print([2,2,2] == (2,2,2))
```

Selecione uma opção de resposta:

- None, there's a syntax error
- None, there's a *runtime* error
- `True`
- `False`

Can I have names in my matrix columns (data frames) using NumPy?

- No, I need to use OpenCV
- Yes
- No, I need to use Pandas
- No, I need to use SymPy

To what can **assert** can be used?

- Ensure the program keeps running even if a variable value is invalid
- To document the code
- Crash the program as soon as a variable value is found to be invalid
- It can be used only for variables

What is the output of the following statements?

```
s = "Gin tonic"
print(s[7:11] * 3)
```

Selecione uma opção de resposta:

- `nicnicnic`
- None, it gives a *runtime* error
- `icicic`
- `nic*3`

What is the output of the following statements?

```
s = "Gin tonic"
print(s[1] * s.index("t"))
```

Selecione uma opção de resposta:

- None, it gives a *runtime* error
- `iiii`
- `GGGG`
- `t`

Which Python code must be used to create the following set?

```
{'S', 'u', p', 'e', 'r', ' ', 'B', 'o', 'c', 'k'}
```

Selecione uma opção de resposta:

- `set{'Super Bock'}`
- `set(['Super Bock'])`
- `set({'Super Bock'})`
- `set('Super Bock')`

Is it advisable to use `print` statements inside a function body?

Selecione uma opção de resposta:

- Yes, they simply automate the manual steps you would take
- No, one must avoid calling `print` inside functions at all times
- No, they simply automate the manual steps you would take
- Yes, temporally during code development to inspect variables

What is the output of the following statements?

```
s = "Gin tonic"
s[4] = "T"
print(s)
```

Selecione uma opção de resposta:

- None, it gives a *runtime* error
- `Gin tonic`
- `T`
- `Gin Tonic`

What is the output of the following statements?

```
tp = ("Gin", "tonic", 0.75)
tp[2] *= 2
print(tp)
```

Selecione uma opção de resposta:

- `('Gin', 'tonictonic', 0.75)`
- `('Gin', 'tonic', 1.5)`
- `("Gin", "tonic", 0.75, "Gin", "tonic", 0.75)`
- None, it produces a *runtime* error

Can I have names in my matrix columns (data frames) using NumPy?

- No, I need to use OpenCV
- Yes
- No, I need to use Pandas
- No, I need to use SymPy

What is the output of the following statements?

```
git = ("Gin", "tonic", 0.75)
print(("Drink",) + git[0:1] + ())
```

Selecione uma opção de resposta:

- `('Drink', 'Gin')`
- None, it produces a *runtime* error
- `'Drink Gin tonic'`
- `('Drink', 'Gin', 'tonic')`

What is the output of the following statements?

```
mpfc = "Monty Python's Flying Circus"
words = mpfc.split()
print(" ".join(words) )
```

Selecione uma opção de resposta:

- `"Monty Python's Flying Circus"`
- `['Monty', "Python's", 'Flying', 'Circus']`
- None, there's a *runtime* error
- `"MontyPython'sFlyingCircus"`

What can SymPy be used for?

- Numerical computations
- User interfaces
- Web servers
- Analytical manipulation

What is the output of the following statements?

```
alist = ["A", "bottle", "gin", [50, "beer"], [], True]
print(alist[4:])
```

Selecione uma opção de resposta:

- `[[24, "beer"], [], True]`
- `[[], True]`
- `True`
- `[24, "beer", True]`

What is PEP8 — Style Guide for Python Code?

Selecione uma opção de resposta:

- The standard rules that Spyder IDE uses when editing Python code
- The standard rules to write *docstrings* in order to better document Python functions
- A set of rules that Python programmers should use to improve readability of the programs
- A set of Python rules that are compulsory during program development

What is the output of the following statements?

```
alist = [1, 3, 2]
print(alist * 2)
```

Selecione uma opção de resposta:

- `[1, 1, 3, 3, 2, 2]`
- `2, 6, 4]`
- `[1, 3, 2, 1, 3, 2]`
- `12`

What is the output of the following statements?

```
numbers = {1, 2, 3, 4, 5, 6}
numbers.discard(0)
print(numbers)
```

Selecione uma opção de resposta:

- `{}`
- `{1, 2, 3, 4, 5, 6}`
- None, there's a *syntactic* error because there's no `remove()` method on set objects
- None, there's a *runtime* error

What is the output of the following statements?

```
alist = [1, 2, 3, 4]
blist = []
for item in alist:
    blist.append(item+2)
print(blist)
```

Selecione uma opção de resposta:

- `[6, 5, 4, 3]`
- `[3, 4, 5, 6]`
- None, there's a *runtime* error (you cannot concatenate inside an append)
- `[1, 2, 3, 4, 2]`

What is the output of the following statements?

```
range(2, 0, -1)
```

Selecione uma opção de resposta:

- `[2, 1]`
- `range(2, 0, -1)`
- `(2, 0, -1)`
- `[2, 0, -1]`

Remember that NumPy does arithmetic broadcasting.
If I do C=A*B, where A.shape is (6, 3) and B.shape is (6,1), what is C.shape?

- (3,6)
- (6,1)
- (6,3)
- There’s a runtime error because shapes do not match

Which datatypes are immutable in Python?

Selecione uma opção de resposta:

- Dictionary and String
- String and Tuple
- String and List
- Set and List

What is the output of the following statements?

```
s = "Gin tonic"
print(s[len(s)-5])
```

Selecione uma opção de resposta:

- `c`
- `t`
- None, it gives a *runtime* error
- `o`

What is the output of the following statements?

```
git = ("Gin", "tonic", 0.75)
print(("Drink",) + git[0:1] + ())
```

Selecione uma opção de resposta:

- `('Drink', 'Gin')`
- None, it produces a *runtime* error
- `'Drink Gin tonic'`
- `('Drink', 'Gin', 'tonic')`

What is the output of the following statements?

```
gt = ("Gin", "tonic", 0.75)
print(gt[1:])
```

Selecione uma opção de resposta:

- `'Gin'`
- `('tonic', 0.75)`
- None, it produces a *runtime* error
- `('Gin')`

In Numpy, if I have a matrix M(6x3), how do I choose the third column?

- M[3]
- M[3, :]
- M[: , -1]
- M[: , 3]

What is the output of the following statements?

```
l1 = [1, 3, 5, 7]
l2 = l1
print(l2 is l1)
```

Selecione uma opção de resposta:

- `False`
- `True`
- None, there's a syntax error
- None, there's a *runtime* error

What is the output of the following statements?

```
mydict = {"Cleese":80, "Idle":76, "Palin":77}
answer = list(mydict.items())[1]
print(answer)
```

Selecione uma opção de resposta:

- `['Idle', 76]`
- None, there's a *runtime* error
- `('Idle', 76)`
- `76`

If I want to work with computer vision, what package should I choose?

- NumPy
- OpenCV
- Pandas
- SymPy

In a function call in Python, can we have another function call as an actual parameter?

Selecione uma opção de resposta:

- Yes, we can if the value the *inner* function returns is a natural number n ≥ 2
- Yes, we can if the *inner* function is a boolean function
- No, we can not
- Yes, we can

Consider the following comprehension:

```
dict = {str(i):2*i for i in range(1, 4)}
dict = {v:k for k,v in dict.items()}
print(dict)
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- None, it gives a *runtime* error
- A dictionary with 4 integer string and 4 integer values
- A list with for 4 pairs (integer, string)
- A dictionary with 4 integer keys and 4 string values

Which of the list operations shown below is not O(1)?

- list[10]
- list.pop()
- list.append()
- list.pop(0)

Consider the following Python code:

```
def outer_function(text):
    itext = text
    def inner_function(text): # nested function
       itext = text + " " + itext
    inner_function('Hello')
outer_function('World')
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- `Hello World`
- `World Hello`
- None, it gives a *runtime* error
- `World World`

In order to obtain the integer equal to the multiplication of the members of the given list, what is the Python function, or object method, that may be used?

Selecione uma opção de resposta:

- `map(operator.mul, [1, 3, 5, 7, 13])`
- `functools.reduce(operator.mul, [1, 3, 5, 7, 13], 1)`
- `curry(operator.mul, 1, [1, 3, 5, 7, 13])`
- `filter(operator.mul, [1, 3, 5, 7, 13])`

Given the following code fragment what is its Big-O running time?

```
k = n
while k > 0:
        j += 2
        k = k // 2
```

- O(n^3)
- O(n)
- O(n^2)
- O(log n)

Given the following code fragment what is its Big-O running time?

```
t = 0
for i in range(n):
        t += 1
for j in range(n):
        t -= 1
```

- O(n^3)
- O(n^2)
- O(n log n)
- O(n)

What is the result of np.median([1, 7, 10])?

- 10
- 6
- 7
- 1

To compute the sum of the integers of the given list, how many recursive calls are made by the function call `recursive_sum([1, [3, 5], [7, 11], 13])`?

```
def recursive_sum(nlist):
   total = 0
   for element in nlist:
      if type(element) == type([]):
         total += recursive_sum(element)
      else:
         total += element
   return total
```

Selecione uma opção de resposta:

- 5
- 2
- 4
- 6

What is the output of the following statements?

```
months = {"Sep", "Oct", "Nov", "Dec", "Jan"}
print(months[1])
```

Selecione uma opção de resposta:

- `['Oct', 'Nov', 'Dec', 'Jan']`
- None, there's a *runtime* error (one cannot index a set)
- `'Oct'`
- `'Sep'`

Consider the recursive implementation in Python of the Tower of Hanoi mathematical game.

If the base case is not reached, how many recursive calls are made inside each activation?

Selecione uma opção de resposta:

- 2
- 3
- 4
- 1

How many lines does the following piece of Python code prints in the standard output?

```
for _, o in [("John", 1), ("Eric", 2), ("Michael", 3)]:
    print("The name is", o)
```

Selecione uma opção de resposta:

- It prints a floating point number
- 3
- 4
- 6

Consider the `for` statement with a display literal:

```
for f,q in [("apples",6), ("bananas",12), ("quinces",18)]:
    print("{0} quantity {1}".format(f, q))
```

How many prints will occur when it is submitted to the interpreter?

Selecione uma opção de resposta:

- none
- 1
- 3
- 6

Consider the `for` statement with a list comprehension:

```
for (f,x) in [(x, 3*x) for x in range(1, 4)]:
   print(f, x)
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- None, it gives a *runtime* error
- it prints a list with 3 pairs with a number and its triple
- It prints 3 pairs with a number and its triple
- It prints 3 lines each with a number and its triple

To compute the sum of the integers of the given list, how many recursive calls are made by the function call `listsum([1, 3, 5, 7, 11, 13])`?

```
def listsum(nlist):
    if len(nlist) == 1:
       return nlist[0]
    else:
       return nlist[0] + listsum(nlist[1:])
```

Selecione uma opção de resposta:

- 3
- 6
- 4
- 5

Consider the following Python code:

```
def outer(a):
    def inner(b):
       return a ** b
    return inner
print(outer(3)(2))
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- 6
- 8
- 9
- None, it gives a *runtime* error

Consider the following piece of Python code:

```
import functools
print(functools.reduce(lambda a, b: a+b, range(5), 0))
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- `5`
- `<range object at 0x7fe21cd42dd8>`
- `10`
- None, it gives a *runtime* error

Consider the following piece of Python code:

```
(lambda n: n*n)(3) + (lambda n: n+n)(3)
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- It prints the string `"3 + 3"`
- It prints the pair `(3,3)`
- None, it gives a *runtime* error
- It prints the integer `15`

Given a function f(x, y), we can define a function g such that g(x)(y) is equivalent to f(x, y). What do we call this transformation that uses HOF?

Selecione uma opção de resposta:

- Uncurrying
- Function composition
- A closure
- Currying

Recursion and iteration perform the same kinds of tasks: solve a complicated task one piece at a time, and combine the results.

What is the emphasis of Recursion?

Selecione uma opção de resposta:

- Solve a large problem by breaking it up into smaller and smaller pieces until it can be solved
- Use dictionaries to capture intermediate results to be very efficient in the partial computations
- Keep repeating until a task is finished and combine the results
- Use an accumulator variable to compute a running total of the computation

Consider the following Python code:

```
def outputter(s):
    return s + "..."
inputter = outputter
print(inputter("Hello"), outputter("World"))
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- `World... Hello...`
- `Hello... World...`
- None, it gives a *runtime* error
- `Hello World....`

Consider the recursive and the recursive with memoisation implementations of Fibonacci in Python.

Which of the following is true?

Selecione uma opção de resposta:

- The version with memoisation does not make any function call
- Both implementations make approximately the same number of calls
- The version with memoisation makes thousands more calls than the recursive version
- The version with memoisation makes thousands less calls than the recursive version

Recursive implementations have advantages and disadvantages over iterative implementations for the same problem.

Which one is an advantages of recursion?

Selecione uma opção de resposta:

- The recursive version is always faster
- Recursion uses less space in activation calls
- Recursion makes debug easier
- Recursive functions go well with tree traversals

Consider the following piece of Python code:

```
heights = [(5,7), (5,8), (5,9), (6,0), (6,1)]
list(filter(lambda x: x[0] > x[1], heights))
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- `2`
- `[(6, 0), (6, 1)]`
- None, it gives a *runtime* error
- `<filter object at 0x7fe21cd42dd8>`

Consider the following piece of Python code:

```
heights = [(5,7), (5,8), (5,9), (6,0), (6,1)]
def convert(ftin):
   feet, inches = ftin
   return 12.0*feet + inches
map(convert, heights)
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- None, it gives a *runtime* error
- It prints `<map object at 0x7fe21cd42d68>`
- It prints a list with 5 pairs with two numbers
- It prints a floating point number

The *Hofstadter Female and Male* sequences are an example of what?

```
F(0) = 1
M(0) = 0
F(n) = n - M ( F ( n - 1 ) ), n > 0
M(n) = n - F ( M ( n - 1 ) ), n > 0
```

Selecione uma opção de resposta:

- Natural recursion
- Tail recursion
- Mutual recursion
- Infinite recursion

Consider a recursive and an iterative implementation in Python of the mathematics factorial number.

Which of the following is true?

Selecione uma opção de resposta:

- The recursive version is easier to read
- The recursive version is easier to trace and debug
- The recursive version is faster
- The recursive version uses less memory

Consider the following list comprehension:

```
[(x, 2*x) for x in range(1, 7) if x % 2 != 0]
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- It prints 6 lines each with an odd number and its double
- It prints 6 pairs with an odd number and its double
- It prints `<tuple object at 0x7fe21cd42d68>`
- It prints a list with 3 pairs with an odd number and its double

What is the last output made by the function call `recursion_depth(5)`?

```
def recursion_depth(number):
  print("{0}, ".format(number), end="")
  recursion_depth(number + 1)
```

Selecione uma opção de resposta:

- `6`
- There's a *runtime* error
- `4`
- `2`

What is the scope of the variable **`range`** if declared outside any of the *script* functions?

Selecione uma opção de resposta:

- [Clear answer]
- There's a syntax error because it is a reserved word
- **`global`**
- **`local`**
- **`built-in`**

Some objects can be compared using ==, but not using > because they have no order.
For example, if `d1` and `d2` are two dictionaries, `d1==d2` is valid, but `d1>d2` is not valid.
For such objects, which search algorithms can be used?

Selecione uma opção de resposta:

- Linear and binary search
- Binary search
- Linear search, binary search and merge sorted
- Linear search

Which algorithm do you choose to find a word in a sorted vocabulary of words?

Selecione uma opção de resposta:

- Remove all adjacent duplicates
- Binary search
- Merge sorted
- Linear search

If you have the sorted list `[3, 5, 6, 8, 11, 12, 14, 15, 19]` and are using the binary search algorithm, which list of numbers correctly shows the sequence of comparisons used to search for the key `7`?

Selecione uma opção de resposta:

- `[11, 6, 8]`
- `[11, 5, 6]`
- `[12, 6, 11, 8]`
- `[8, 5, 6]`

Consider the following extract of Python code:

```
def mk_mult(n):
    return lambda x: n * x
f = mk_mult(2)
f(5)
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- `<function <lambda> at 0x7ff19901c9d8>`
- `5`
- None, there's a *runtime* error
- `10`

Why does a programmer uses files?

Selecione uma opção de resposta:

- To achieve persistence of the data that the program manipulates
- To synchronise the data the program manipulates with data in RAM
- To manipulate data in RAM memory
- [Clear answer]
- To manipulate data in volatile memory

The merge sorted algorithm merges two lists into a single sorted list. What are the characteristics of the initial lists?

Selecione uma opção de resposta:

- The first argument list must be sorted but there's no requirement in the second argument list
- Nothing special as the algorithm simply concatenates the two lists together and sorts the result
- Both argument lists must be sorted
- Nothing special as the resulting elements are inserted in sort order by the algorithm

Which is the statement that a Python programmer uses to *throw* a *runtime* exceptions to signal a *runtime* error?

Selecione uma opção de resposta:

- `throw <exception>`
- [Clear answer]
- `raise <exception>`
- `try <exception>`
- `except <exception>`

When is the code inside a **`finally`** block executed in Python?

Selecione uma opção de resposta:

- It is executed when a *runtime* exception occurs
- It is executed when there's the "unwinding of the stack" after a **`raise`** statement
- It is always executed
- It is executed when there's no *runtime* exception
- [Clear answer]

Which is the statement that a Python programmer uses when she wants to state an *invariant* at *runtime* about a function argument type?

Selecione uma opção de resposta:

- `assert`
- `try ... catch`
- `invariant`
- [Clear answer]
- `finally`

Consider the following extract of Python code:

```
with open("test.txt", "r") as f:
    content = f.readlines()
```

If the file `test.txt` exists in the current directory, what is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- All content of the file is bound to the list variable `content`
- Syntax error because because `readlines()` is not a method of handler `f`
- *Runtime* error
- [Clear answer]
- All content of the file is bound to the string variable `content`

What does a Python programmer use to focus on the values of a list (What) rather than on the procedure to manipulate a list (How)?

Selecione uma opção de resposta:

- `for` statements over lists
- `while` statements and modifiers on lists
- `list()` statement
- List comprehensions

Consider a map of function `f` to list `l`:

```
map(f, l)
```

What is this map equivalent to?

Selecione uma opção de resposta:

- `(x for x in l if f(x))`
- `[x for x in l if f(x)]`
- `(f(x) for x in l)`
- `[f(x) for x in l]`

To execute code in the `main()` function only when a module executes by itself in a standalone fashion, what is the variable that must be used in the test?

Selecione uma opção de resposta:

- [Clear answer]
- `__main__`
- `import`
- `__name__`
- `namespace`

Consider the following extract of Python code:

```
def f(lst, e):
    lst.append(e)
```

How do you classify the function `f()`?

Selecione uma opção de resposta:

- A function generator
- An higher-order function
- A modifier
- A pure function

What is the Python `else` statement used for?

Selecione uma opção de resposta:

- To have code in a `try` statement that executes when a *runtime* exception occurs
- To have code in a `try` statement that executes instead of the `if` statement
- To have code in a `try` statement that executes when there's no `finally` statement
- To have code in a `try` statement that executes when there's no *runtime* exception
- [Clear answer]

Consider the following extract of Python code:

```
r = range(5, 10)
next(r)
```

What is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- None, there's a *runtime* error
- `5`
- `0`
- `6`

Consider the following extract of Python code:

```
with open("test.txt", "r+") as f:
    f.write("My first file written from Python\n")
```

When it is submitted to the interpreter, if it succeeds to open the file `test.txt`, what is the output?

Selecione uma opção de resposta:

- The string shown is read from the disk
- The string shown is written to the disk
- The string shown is written to the disk and appended to the file
- [Clear answer]
- *Runtime* error

Consider the following extract of Python code:

```
t0 = time.clock()
```

When it is submitted to the interpreter what is the value of variable `t0`?

Selecione uma opção de resposta:

- The number of milliseconds since the program started
- [Clear answer]
- The number of seconds since the program started
- The number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC)
- The integer `0`

Consider the following extract of Python code:

```
with open("test.txt") as f:
    content = f.read()
```

If the file **`test.txt`** exists in the current directory, what is the output when it is submitted to the interpreter?

Selecione uma opção de resposta:

- *Runtime* error
- All content of the file is bound to the string variable **`content`**
- All content of the file is bound to the list variable **`content`**
- [Clear answer]
- Syntax error because the *mode* for open is missing in the call

Consider the following extract of Python code:

```
import random
rng = random.Random()
numbers = rng.randrange(1, 7)
```

When it is submitted to the interpreter what is the value of variable **`numbers`**?

Selecione uma opção de resposta:

- [Clear answer]
- A real number ∈ [1, 7[
- One of 1, 2, 3, 4, 5, 6, 7
- One of 1, 2, 3, 4, 5, 6
- A real number ∈ [1, 7]

Consider the following extract of Python code:

```
with open("test.txt", "r") as f:
    f.write("My first file written from Python\n")
```

When it is submitted to the interpreter, if it succeeds to open the file `test.txt`, what is the output?

Selecione uma opção de resposta:

- The string shown is written to the disk
- *Runtime* error
- [Clear answer]
- The string shown is written to the disk and appended to the file
- The string shown is read from the disk

In linear search, what is the worst case scenario?

Selecione uma opção de resposta:

- We have to to go about halfway through the list doing N/2 probes
- The worst case is when the list is not ordered
- It is not possible to determine, as it depends on the type of the elements of the list
- We have to probe every one of the N elements of the list

Which is the statement that a Python programmer uses to catch and handle *runtime* exceptions that may occur during program execution?

Selecione uma opção de resposta:

- `try ... except`
- `try ... finally`
- [Clear answer]
- `try ... catch`
- `try ... else`

What needs to be done to create a new module?

Selecione uma opção de resposta:

- Use the statement: **`import module`**
- Save the *script* in a file name with a **`.py`** extension
- Use the statement: **`def module`**
- [Clear answer]
- A regular programmer cannot create new modules