Question 1

Incorrect Note de -0,20 sur 1,00

^[^x] returns exactly (i.e. returns those and no other) the lines that ...

(Bareme: -20% penalty if incorrect selection)

Veuillez choisir une réponse.

- a. start with an hexadecimal number
- b. start with x
- c. are non-empty and do not start with x
- d. contain any string but x
- e. I don't know (no points, no penalty).
- [✓ f.] do not start with x, whether they are empty or not. ✖

Votre réponse est incorrecte.

La réponse correcte est : are non-empty and do not start with x

Question 2

Correct Note de 1,00 sur 1,00

grep -E '^$' f

returns:

(Bareme: -16.67% penalty if incorrect selection)

Veuillez choisir une réponse.

- a. non-empty lines of file f
- b. I don't know (no point, no penalty)
- [✓ c. all empty lines of file f] ✓
- d. all lines of file f

- e. the first non-empty line of file f
- f. all lines starting with $ in file f
- g. nothing
- h. the first line starting with $ in file f

Votre réponse est correcte.

La réponse correcte est : all empty lines of file f

Question 3

Partiellement correct Note de 0,40 sur 1,00

Indicate what is returned by **re.search('z*c?', 'bzcd').span()**

We recall that the Python function re.search(expression, text) returns the first match of expression in text (None if there is no match). The semantics adoptes is the PCRE one.
Applied to a match object, the span() method returns the pair of integers (i,j) such that the match is text[i]...text[j-1].
Par exemple, **re.search('aa', 'bcdaa').span()** renvoie **(3, 5)**.

Bareme: up to -20% penalty if incorrect selection.

○ a. I don't know (no point, no penalty)

- b. (2, 4)
- c. (0, 0)
- [✓ d. (1, 3)] ✓
- e. (3, 5)
- f. (0, 2)
- g. None

Votre réponse est partiellement correcte.

La réponse correcte est :
(0, 0)

Question 4

Correct Note de 1,00 sur 1,00

Indicate which expression will replace each digit, letter and underscore character of string '**aB cde g 2**' with a * in the following code:

re.sub(expression, r'*', 'aB cde g 2') # should thus return **'** *** * *'**

We recall that the Python function re.sub(expression, repl, text) replaces the (non-overlapping) matches for the expression in text by the replace string (characters which are not part of the match are kept as-is).
The semantics adopted in Python is PCRE.
For instance: re.sub('a', 'z', 'abacaad') renvoie 'zbzczzd'

Bareme: up to -20% penalty per incorrect selection.

[✓ a.] **expression = r'\w'** *#(w is lowercase here)* ✓

b. **expression = r'\w*'** *#(w is lowercase here)*

c. **expression = r'\W*'** *#(w is uppercase here)*

d. **expression = '[a-z]'**

e. **expression = r'\w+'** *#(w is lowercase here)*

f. **expression = r'\W+'** *#(w is uppercase here)*

g. **expression = r'\W'** *#(w is uppercase here)*

h. **expression = r'[a-z][0-9]'**

i. I don't know (no point, no penalty)

Votre réponse est correcte.