python_basic_programming_18

May 20, 2023

[ ]:

```
1.Create a function that takes a list of non-negative integers and strings and␣
 ↪return a new list without the strings ?
Examples:
filter_list([1, 2, "a", "b"]) □ [1, 2]
filter_list([1, "a", "b", 0, 15]) □ [1, 0, 15]
filter_list([1, 2, "aasf", "1", "123", 123]) □ [1, 2, 123]
```

[1]:

```
def filter_list(list):
    out_string = []
    for ele in list:
        if type(ele) == int and ele >= 0:
            out_string.append(ele)
    return out_string

print(f'  {filter_list([1, 2, "a", "b"])}')
print(f'  {filter_list([1, "a", "b", 0, 15])}')
print(f'  {filter_list([1, 2, "aasf", "1", "123", 123])}')
```

```
  [1, 2]
  [1, 0, 15]
  [1, 2, 123]
```

[ ]:

```
2. The "Reverser" takes a string as input and returns that string in reverse␣
 ↪order, with the opposite case ?
Examples:
reverse("Hello World") □ "DLROw OLLEh"
reverse("ReVeRsE") □ "eSrEvEr"
reverse("Radar") □ "RADAr"
```

[2]:

```
def reverse(in_string):
    print(f'{in_string}  {in_string[::-1].swapcase()}')

reverse('Hello World')
reverse("ReVeRsE")
reverse("Radar")
```

```
Hello World  DLROw OLLEh
ReVeRsE  eSrEvEr
Radar  RADAr
```

[ ]:

```
3.You can assign variables from lists like this:
lst = [1, 2, 3, 4, 5, 6] first = lst[0] middle = lst[1:-1] last = lst[-1]␣
 ↪print(first) □
outputs 1 print(middle) □ outputs [2, 3, 4, 5] print(last) □ outputs 6
With Python 3, you can assign variables from lists in a much more succinct way.
Create variables first, middle and last from the given list using destructuring␣
 ↪assignment
(check the Resources tab for some examples), where:
first □ 1 middle □ [2, 3, 4, 5] last □ 6
Your task is to unpack the list writeyourcodehere into three variables,
being first, middle, and last, with middle being everything in between
the first and last element. Then print all three variables.
```

[3]:

```
first, *middle, last = [1,2,3,4,5,6]
print(f'first   {first}')
print(f'middle   {middle}')
print(f'last   {last}')
```

```
first   1
middle   [2, 3, 4, 5]
last   6
```

[ ]:

```
4.Write a function that calculates the factorial of a number recursively.
Examples:
factorial(5) □ 120
factorial(3) □ 6
factorial(1) □ 1
factorial(0) □ 1
```

[4]:

```
def factorial(n):
    if n==0:
        return 1
    return n * factorial(n-1)

print(f'factorial(5)   {factorial(5)}')
print(f'factorial(3)   {factorial(3)}')
print(f'factorial(1)   {factorial(1)}')
print(f'factorial(0)   {factorial(0)}')
```

```
factorial(5)   120
factorial(3)   6
factorial(1)   1
factorial(0)   1
```

[ ]:

```
5.Write a function that moves all elements of one type to the end of the list.
Examples:
move_to_end([1, 3, 2, 4, 4, 1], 1) □ [3, 2, 4, 4, 1, 1]
# Move all the 1s to the end of the array.
```

```
move_to_end([7, 8, 9, 1, 2, 3, 4], 9) ▯ [7, 8, 1, 2, 3, 4, 9]
move_to_end(["a", "a", "a", "b"], "a") ▯ ["b", "a", "a", "a"]
```

[5]:
```
def move_to_end(list,num):
    first_end = []
    second_end = []
    for ele in list:
        if ele == num:
            second_end.append(ele)
        else:
            first_end.append(ele)
    first_end.extend(second_end)
    return first_end

print(f'move_to_end([1, 3, 2, 4, 4, 1], 1)   {move_to_end([1, 3, 2, 4, 4, 1],␣
 ↪1)}')
print(f'move_to_end([7, 8, 9, 1, 2, 3, 4], 9)   {move_to_end([7, 8, 9, 1, 2, 3,␣
 ↪4], 9)}')
print(f'move_to_end(["a", "a", "a", "b"], "a")   {move_to_end(["a", "a", "a",␣
 ↪"b"], "a")}')
```

```
move_to_end([1, 3, 2, 4, 4, 1], 1)   [3, 2, 4, 4, 1, 1]
move_to_end([7, 8, 9, 1, 2, 3, 4], 9)   [7, 8, 1, 2, 3, 4, 9]
move_to_end(["a", "a", "a", "b"], "a")   ['b', 'a', 'a', 'a']
```