

MOCK TEST3

(p1.py) Create a function `f(word)` that, for a given word `word`, returns a string of characters in which the successive letters of the word create a so-called Mexican wave. Initially, the first letter of the word is uppercase and the remaining letters are lowercase. Then the second letter of the word is uppercase and the remaining letters are lowercase, etc. Separate the words with a minus sign (-). Example:

```
f("water") returns "Water-wAter-waTer-watEr-waterR"
f("a") returns "A"
f("") returns ""
```

(p2.py) Class `C` describes a point `(x,y)` in the plane. The point coordinates are given when creating (initializing) the object. The class contains the `m1()` method that returns the number of the quadrant of the Cartesian system in which the point `(x,y)` is located ([https://en.wikipedia.org/wiki/Quadrant_\(plane_geometry\)](https://en.wikipedia.org/wiki/Quadrant_(plane_geometry))). The `m1()` method returns 0 if the point `(x,y)` is located on the X-axis or Y-axis. The class contains the `m2(a,b)` method that returns true when the point `(x,y)` is in the same quadrant of the Cartesian system as the point with coordinates `a,b`. Otherwise, the method returns false. The class contains the `m3(a,b)` method that returns true when the distance between points `(x,y)` and `(a,b)` is greater than 5. Otherwise, the method returns false. Example:

```
p = C(2,3)
p.m1() returns 1
p.m2(7,4) returns True
p.m2(-3,1) returns False
p.m3(8,5) returns True
p.m3(4,7) returns False
p1 = C(0,5)
p1.m1() returns 0
p1.m2(4,7) returns False
p1.m2(-7,0) returns True
```

(p3.py) Flight numbers along with the number of passengers are stored in a dictionary `d`. Define a function `f(d)` that returns the number of flights in which the number of passengers is greater than the average number of passengers on all flights. Example:

```
f({"LO231":150,"BA787":120,"NZ15":30}) returns 2
f({"LO231":150,"BA787":20,"NZ15":30}) returns 1
```

(p4.py) The `res` array contains test results, i.e. the number of points between 0 and 100. Create a function `f(fnc,res)` that filters the test results according to the criteria contained in the `fnc` function. The `f` function returns the difference between the highest and lowest test result. Example:

```
res = [95,90,20,50,70]
fnc1 = lambda x: x>50
f(fnc1,res) returns 25
fnc2 = lambda x: x>30 and x<90
f(fnc2,res) returns 20
```

(p5.py) A valid number on the planet Metis consists of the digits 1 through 7 and the lower or upper case letters a through d. A plus (+) or minus (-) sign can also appear at the beginning of the number. Create a function `f(mnumbers)` that returns how many numbers in the array `mnumbers` are valid on the planet Metis. Example:

```
f(["A15", "-31", "7abC", "+D1", "-g4"]) returns 4
f(["A05", "-3+1", "7ab8C", "+Bb7", "-22c55"]) returns 2
```

(p6.py) Define a class `C` that allows you to create objects representing people. The class constructor has 3 parameters: first name, last name, and age. The text representation of the object contains the initial of the first name and last name and the age of the person. If the person age is less than 18 years old, their initials consist of lowercase letters, and if they are an adult, they consist of uppercase letters. Example:

```
C("John", "May", 21) returns "JM21"
C("Anna", "Brown", 17) returns "ab17"
```

(p7.py) A counter allows you to count arbitrary elements. Define a class `C` that allows you to create counters. The initial value of the counter is assigned when the object is created. The class contains the following methods: `m1()` (returns the value of the counter), `m2()` (increments the value of the counter by 1), `m3()` (decrements the value of the counter by 1), `m4(n)` (increments or decrements the value of the counter by `n`), `__str__()` (returns the value of the counter as a string). Example:

```
c=C(5)
c.m1() returns 5
c.m2()
c.m1() returns 6
c.m4(-8)
c.m1() returns -2
c.m3()
c.m1() returns -3
c.m4(10)
c.m1() returns 7
c.__str__() returns "7"
```

(p8.py) The `prods` array contains the names of products in stock. Create a function `f(fnc,prods)` that maps product names to their IDs, as defined by the `fnc` function. The `f` function returns a comma-separated text string of product IDs, without spaces. Example:

```
prods = ["water", "cheese", "tomato"]
fnc1 = lambda x: "id:"+x[:2]
f(fnc1,prods) returns "id:wa,id:ch,id:to"
fnc2 = lambda x: (x[0]+x[-1]).upper()
f(fnc2,prods) returns "WR,CE,TO"
```

(p9.py) The `uid` array contains sample user IDs for a popular website. The IDs must be unique. Create a function `f(uid)` that returns true if all the given IDs are unique. Otherwise, the function returns false. Example:

```
f(["john5", "ann123", "JOHN5", "xxx", "abc333", "a10"]) returns True
f(["abc123", "ann", "abc123", "a10"]) returns False
```

(p10.py) Create a function `f(value1)` that returns a function with one parameter `value2` of integer type. The returned function calculates and returns the product `value1 * value2`. Example:

```
times_five = f(5)
times_five(8) returns 40
times_three = f(3)
times_three(7) returns 21
```

(p11.py) The use of company cars is available in the form of a list (data structure: array of dictionaries) containing the vehicle registration number and the number of kilometers traveled. Create a function `f(car,order)` that returns a list of cars sorted alphabetically (when `order = 1`) or a list of cars sorted by the number of kilometers traveled, in descending order (when `order = 2`). Example:

```
cars = [{"KR333":138}, {"WL555":497}, {"DB444":341}, {"MC222":412}]
f(cars,1) returns [{"DB444":341}, {"KR333":138}, {"MC222":412}, {"WL555":497}]
f(cars,2) returns [{"WL555":497}, {"MC222":412}, {"DB444":341}, {"KR333":138}]
```

(p12.py) The ISO date format is a standard that provides a unique way to represent dates, independent of local conventions. An ISO date is represented in the form YYYY-MM-DD, where YYYY is the year, MM is the month, and DD is the day, such as 2025-01-04 for January 4, 2025. Create a function `f(dates)` that, given a string of dates separated by a comma, returns an array of dates that conform to the ISO format. Example:

```
dates = "2021-1-3,05/12/2024,1998-12-11,9 maj 2007,2001-12-07,15-09-2011"
f(dates) returns ["1998-12-11", "2001-12-07"]
```