# Mock2 Test3

NAME, SURNAME, STUDENT ID (IN CAPITAL LETTERS): ……………………………………………………………………………………………

**Create programs in files with the names given in parentheses at the beginning of each task. If you use other names, you will not receive points.**

**Each function has a name f(). If you use a different name, you will not receive points.**

(p1.py) Create a function f(n) that returns the difference between the largest and smallest odd digit contained in the number n. When the number n does not contain odd digits, the function returns -1. Example:

f(10852) 🡪 4  
f(7235973) 🡪 6  
f(4388) 🡪 0  
f(846206) 🡪 -1

(p2.py) Create a function f(x,y,d) that returns true when the string of digits d appears in any number between x and y. Otherwise, the function returns false. Example:

f(10,15,"14") 🡪 True  
f(100,120,"11") 🡪 True  
f(205,210,"04") 🡪 False

(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}) 🡪 2  
f({"LO231":150,"BA787":20,"NZ15":30}) 🡪 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) 🡪 25  
fnc2 = lambda x: x>30 and x<90  
f(fnc2,res) 🡪 20

(p5.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)> ). 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() 🡪 1  
p.m2(7,4) 🡪 True  
p.m2(-3,1) 🡪 False  
p.m3(8,5) 🡪 True  
p.m3(4,7) 🡪 False  
p1 = C(0,5)  
p1.m1() 🡪 0  
p1.m2(4,7) 🡪 False  
p1.m2(-7,0) 🡪 True

(p6.py) A valid number on the planet Metis consists of digits 1 to 7 and lowercase or uppercase letters a to d. A plus (+) or minus (-) sign may also appear at the beginning of the number. The mnumbers array contains sample numbers. Create a function f(mnumbers) that returns how many numbers in the array that are valid in the planet Metis. Example:

f(["A15","-31","7abC","+D1","-gH"]) 🡪 5  
f(["A05","-3+1","7ab8C","+D1","-22k"]) 🡪 1

(p7.py) A computer system registers all entries into the car park ("in") and exits from the car park ("out"). Define a function f(d) that for the registered data d returns an array containing the registration numbers of vehicles that remain in the car park, in alphabetical order. Example:

cars = [["KR234","in"],["BA123","in"],["GX444","in"],["KR234","out"], ["BA111","in"],["BA123","out"],["KR234","in"]]  
f(cars) 🡪 ["BA111","GX444","KR234"]  
cars1 = [["KR234","in"],["KR234","out"]]  
f(cars1) 🡪 []