Mock Test3

Wpisz swoje dane personalne:

Imię i nazwisko: ………………………………………………………………………………………………………..

Nr legitymacji (albumu): …………………………………………………………………………………………….

Podpis: ……………………………………………………………………………………………………………………..

Na kolokwium zgłoś się kilka minut przez godziną jego rozpoczęcia. Przynieś legitymację studencką i długopis (pisak, pióro). Wyłącz telefon komórkowy oraz nie korzystaj z żadnych pomocy podczas kolokwium. Jeśli zakończysz kolokwium przed czasem, możesz opuścić salę komputerową.

Wykonaj poniższe zadania. Na ich wykonanie masz 80 minut. Rezultaty zapisz do plików o nazwach podanych w nawiasach. Pliki prześlij na platformę Moodle.

Poza definicją funkcji, nie umieszczaj w plikach innych instrukcji. Aby sprawdzić działanie funkcji, możesz np. utworzyć dodatkowy program Test.py, w którym zaimportuj utworzone moduły i wywołaj zdefiniowane funkcje. Pliku Test.py nie przesyłaj na platformę Moodle.

**UWAGA. Zadania sprawdzane będą automatycznie. Upewnij się, że nazwy utworzonych plików oraz funkcji są zgodne z treścią zadania. W przypadku niezgodności, nie otrzymasz punktów za wykonane zadanie.**

(F1.py) Zdefiniuj klasę C, która zawiera atrybuty name oraz surname. Wartości początkowe atrybutów przekazywane są poprzez parametry konstruktora. Określ tekstową reprezentację obiektu, składającą się z pierwszych liter imienia oraz nazwiska. Inicjały pisane wielkimi literami. Przykład:  
C("anna","may") => "AM"

(F2.py) Zdefiniuj klasę C z metodą m(), która zwraca prawdę, gdy atrybut tekstowy klasy jest isogramem lub fałsz w przeciwnym wypadku. Wartość atrybutu tekstowego przekaż poprzez parametr konstruktora. Isogram to ciąg znakowy, w którym wszystkie znaki są różne. Przykład:  
C("red sun").m() => True  
C("blue water").m() => False  
C("BLUE water").m() => True

(F3.py) Zdefiniuj klasę C z atrybutem zawierającym imiona osób zapisanymi w tablicy. Wartość początkowa atrybutu przekazywana jest poprzez parametr konstruktora. Dodaj w klasie metodę m(), która zwraca prawdę, gdy imiona się powtarzają lub fałsz w przeciwnym wypadku. Przykład:  
C(["Anna"]).m() => False  
C(["Anna","John"]).m() => False  
C(["Anna","John","Anna"]).m() => True  
C(["ANNA","John","Anna"]).m() => True

(F4.py) Zdefiniuj klasę C, która zawiera słownik z wartością zapisaną w systemie pozycyjnym (od dwójkowego do dziesiętnego). Słownik przekazywany jest poprzez parametr konstruktora. Metoda m() zwraca wartość w systemie dziesiętnym lub -1, gdy podana w słowniku wartość jest niegodna z podanym systemem liczbowym. Przykład:  
C({"system":"2”, "value":"101"}).m() => 5  
C({"system":"8", "value":"70281"}).m() => -1

(F5.py) Zdefiniuj klasę C z atrybutem tekstowym t. Wartością atrybutu t jest dowolny tekst przekazany poprzez parametr konstruktora. Metoda m1() zwraca słownik zawierający liczbę wystąpień poszczególnych znaków w tekście t. Metoda m2(c1) zwraca słownik zawierający liczbę wystąpień znaków wymienionych w c1. Przykład:  
C("my moon").m1() => {"m":2,"y":1," ":1,"o":2, "n":1}  
C("my moon").m2("mn") => {"m":2,"n":1}

(F6.py) >= Zdefiniuj klasę C, która zawiera tablicę wartości logicznych. Tablica przekazana jest poprzez parametr konstruktora. Metoda m() zwraca liczbę elementów tablicy, dla których obydwa sąsiadujące elementy mają wartość przeciwną. Przykład:  
C([True,False,False]).m() => 0  
C([True,False,True,False]).m() => 2  
C([True,False,True,True,False,True,False,True,False]).m() => 5

(F7.py) Zdefiniuj klasę C, która umożliwia przetwarzanie dokumentu mockdata.json. Dodaj metodę m(n1,n2), która zwraca liczbę rodzin spełniających kryteria: wiek żony co najmniej n1 lat oraz liczba dzieci co najmniej n2. Dodaj metodę m2(n1), która utworzy plik mockdata1.json, zawierający wykaz rodzin, które posiadają co najmniej n1 dzieci.

(F8.py) Zdefiniuj klasę C, która umożliwia przetwarzanie dokumentu mockdata.xml. Dodaj metodę m(n1,n2), która zwraca liczbę rodzin spełniających kryteria: wiek żony co najmniej n1 lat oraz liczba dzieci co najmniej n2.