Potrzebne pliki są w folderze “pliki”

Lab 2

1. Proszę napisać funkcję sprawdzającą czy liczba przekazana jako parametr jej wywołania jest liczbą pierwszą (1.5p)
2. Proszę utworzyć słownik 50 elementowy, w którym klucze są wartościami losowymi z przedziału [0,100); jeżeli w słowniku istnieje już klucz równy wylosowanej wartości nic nie robimy w przeciwnym wypadku dodajemy do słownika element (klucz: ***isPrime***), gdzie ***isPrime***=True/False (1.5p)
3. Proszę utworzyć listę 100 wartości losowych z przedziału [0,11). Następnie proszę na jej podstawie utworzyć słownik, w którym kluczami będą wylosowane liczby a wartościami z nimi związanymi - listy z indeksami wystąpień tych liczb we wcześniej utworzonej liście (określonymi z wykorzystaniem metody ***index***) (2p)
4. Proszę utworzyć słownik o rozmiarze równym wartości pierwszego parametru przekazanego do programu. Kluczami w tym słowniku mają być kolejne wartości naturalne, a wartościami liczby losowe z przedziału [2,15). Następnie proszę na jego podstawie utworzyć listę, której elementami są krotki (wartość, klucz) oraz słownik, w którym zostaną zamienione miejscami wartości z kluczami (1.5p)
5. Proszę utworzyć listę stu wartości losowych z przedziału [0,20). Następnie na podstawie tej listy proszę utworzyć dwa słowniki: pierwszy - klucze wartości parzyste z listy, drugi - nieparzyste; w obu wartości - lista indeksów (określonych iteracyjnie z wykorzystaniem funkcji ***enumerate*** i metody ***setdefault***) danego klucza w liście.
   Następnie na podstawie jednego z otrzymanych słowników proszę utworzyć nowy słownik, w którym klucze zostają takie same jak w słowniku wejściowym, wartości tworzymy natomiast w sposób następujący: jeżeli lista będąca wartością związaną z danym kluczem zawiera jakieś wartości podzielne bez reszty przez 3 pozostawiamy ją jako wartość związaną z danym kluczem w nowym słowniku, jeżeli takich liczb nie ma - wartość jest krotką: (maksymalny indeks[, minimalny indeks]) (2p)
6. Proszę utworzyć dwa słowniki o rozmiarze 10, w których kluczami są kolejne wartości naturalne, a wartościami liczby losowe z przedziału [1,100). Następnie w obu słownikach proszę zamienić miejscami klucze z wartościami. Na podstawie tak otrzymanych słowników proszę utworzyć nowy słownik, w którym klucze są kluczami występującymi w obu wcześniej utworzonych słownikach, wartościami natomiast są dwuelementowe krotki wartości związanych z danym kluczem w słownikach oryginalnych (1.5p)

Lab3

1. Proszę utworzyć funkcję przyjmującą jeden parametr i zwracającą słownik. Liczba elementów w słowniku jest wartością losową z przedziału [0,20). Kluczami są wartości rzeczywiste z przedziału [0, 1), a wartościami są wartości wyrażenia matematycznego z jedną zmienną przekazanego jako parametr wywołania funkcji obliczone dla danego klucza. Wartości w słowniku mają być zaokrąglone do trzeciego miejsca po przecinku i zapisane z taką dokładnością jako ciągi znaków. Oprócz funkcji ***eval*** można skorzystać z metody ***format*** dla stringów: np. 'x={}'.format(x) (1p)
2. Proszę napisać funkcję, do której można przekazać dowolną liczbę list i krotek a zwracającą listę zawierającą elementy wspólne dla wszystkich parametrów. Proszę użyć konstrukcji for-else (1.5p)
3. Proszę napisać funkcję przyjmującą dwie sekwencje i parametr z wartością domyślną True. Funkcja zwraca listę zawierającą dwuelementowe krotki zawierające elementy o jednakowych indeksach z obu sekwencji. Jeżeli wartość trzeciego parametru wynosi True, długość zwracanej listy równa jest długości krótszej z przekazanych sekwencji, w przeciwnym wypadku brakujące wartości uzupełniamy wartością None. Budowanie każdej z wynikowych list jedna linijka! (2p)
4. Proszę napisać funkcję przyjmującą zmienną liczbę argumentów, która będzie zwracała najmniejszą/największą wartość. Następnie proszę napisać funkcję, przyjmującą jako parametry funkcję oraz zmienną liczba parametrów pozycyjnych. W wywołaniu proszę przekazać funkcję szukającą min/max oraz listę (1.5p)
5. Proszę napisać funkcję umożliwiającą rozmienienie kwoty pieniędzy przekazanej jako jej pierwszy parametr nominałami określonymi poprzez drugi parametr - wartość domyślna krotka (10,5,2) (algorytm zachłanny). Proszę sprawdzić działanie funkcji przekazując inny zestaw monet (2p)
6. Proszę napisać funkcję przyjmującą cztery parametry: liczba, której wartość zgadujemy, granice przedziału, w którym szukana liczba się mieści i ostatni określający sposób poszukiwania wartości z wartością domyślną 'r'. Przy wartości domyślnej ostatniego parametru, liczby poszukujemy losując kolejną wartość, w innym przypadku poszukujemy wartości poprzez podział przedziału poszukiwania wartości na pół. W obu przypadkach w każdym kroku odpowiednio zawężamy przedział poszukiwania (proszę wykorzystać operator trójargumentowy). Proszę sprawdzić ile kroków jest potrzebnych do znalezienia szukanej wartości w zależności od metody (2p)

Lab4

1. Proszę napisać program testujący alternatywne sposoby budowania zestawu wartości: pętla for, lista składana, funkcja map i wyrażenie generatorowe (składnia taka jak listy składanej tylko w miejsce nawiasów kwadratowych należy wstawić okrągłe; o generatorach będziemy mówić na kolejnych zajęciach). Dla każdego ze sposobów proszę utworzyć osobną funkcję tak, aby uzupełnić poniższy kod:

   ```
   import time
   import sys

   powt=1000
   N=10000
   (...)
   print(sys.version)
   test=(forStatement, listComprehension, mapFunction, generatorExpression)
   for testFunction in test:
       print(testFunction.__name__.ljust(20), '=>', tester(testFunction))
   ```

   gdzie: tester - funkcja wywołująca powt razy daną funkcję, w której tworzonych jest N wartości.

   Proszę wykonać testy (wszystko w ramach tych samych funkcji):
   - dodawanie elementu
   - dodawanie elementu podniesionego do kwadratu
   - sumowanie elementów z wykorzystaniem pętli for
   - sumowanie z wykorzystaniem funkcji sum
   - konwersja obiektu map i generatora do listy

   Do pomiaru czasu proszę użyć funkcji time_ns z modułu time. Otrzymane wyniki proszę dołączyć do wysyłanego programu (2p)

2. Proszę utworzyć dwie listy po sto wartości losowych z przedziału [0,20) każda. Następnie na ich podstawie proszę utworzyć listę dwuelementowych krotek, elementów o jednakowych indeksach w listach wyjściowych spełniających warunek, że suma ich wartości jest większa od 3 i mniejsza od 15. Należy wykorzystać listę składaną oraz funkcje ***filter*** i ***zip*** (2p)

3. Proszę napisać funkcję przyjmującą dwa parametry - lista x-ów i y-ów. Korzystając z funkcji wbudowanych ***sum*** i ***map*** proszę obliczyć (i zwrócić z funkcji) wartości dofitowanych współczynników prostej oraz ich niepewności (wzory w pliku) (2p).

4. Proszę napisać funkcję ***myreduce*** przyjmującą dwa parametry (funkcję i sekwencję) oraz zwracającą liczbę. Funkcja przekazywana jako parametr będzie funkcją przyjmującą dwa parametry. Działanie funkcji proszę przetestować korzystając z wyrażenia ***lambda*** dla dodawania i mnożenia (2p)

5. Mamy listę, której elementami są listy dwuelementowe (możemy je potraktować jako współrzędne punktów na płaszczyźnie). Chcemy utworzyć nową listę, w której pierwszym elementem jest lista x-ów, a drugim lista y-ów. Proszę to zrobić w jednej linijce korzystając z funkcji ***myreduce***, wyrażenia ***lambda*** oraz wbudowanej funkcji ***map (obie listy tworzymy jednocześnie!)*** (2p)

Lab5

1. Proszę napisać trzy funkcje generatorowe:
    - zwracającą kolejną liczbę naturalną (nieskończony),
    - zwracającą te wartości z przekazanej jako parametr sekwencji, które są liczbami doskonałymi (liczby naturalne, która są sumą wszystkich swoich dzielników właściwych)
    - zwracającą wartości z przekazanej jako pierwszy parametr sekwencji i przerywającą działanie po napotkaniu wartości większej niż drugi parametr przekazany do funkcji
2.

    Korzystając ze zdefiniowanych funkcji proszę wypisać doskonałe liczby naturalne mniejsze od 10000 (2p)

3. Proszę napisać generator obliczający u_i wg zależności:
$u_i=u_{i-1}+a/x_{i-1}$, z wartością początkową $u_0=0$ dla $x_0=1$ oraz z $x_i=x_0+ia$
Obliczenia proszę wykonać dla a=0.05 i przerwać je dla x=1.5. Zależność pozwala na wyznaczenie przybliżonej wartości logarytmu naturalnego z danej liczby. Generator ma zwracać ***x*** oraz przybliżoną i dokładną wartość logarytmu naturalnego (2p)

4. Każdą liczbę całkowitą można zapisać jako sumę wartości całkowitych mniejszych od niej samej, np. 4 można zapisać jako: 1+1+1+1, 1+1+2, 1+3 oraz 2+2. Proszę napisać generator zwracający wszystkie możliwe sumy dla

określonej wartości ***n*** (2p)

5. Proszę napisać generator zwracający liczby spełniające warunek, że wartość kolejna jest co najmniej o 0.4 mniejsza lub większa od wartości poprzedniej. Działanie generatora należy zakończyć, jeżeli wylosowana wartość jest mniejsza od 0.1 (2p)

6. Proszę napisać generator działający dokładnie tak samo jak wbudowany ***range*** (proszę się upewnić, że wiecie Państwo jak on działa!), ale pozwalający na generowanie liczb rzeczywistych (2p)
   Do testów: range(8), range(-8), range(1,8), range(8,1), range(1,8,2), range(1,8,-2), range(8,1,2), range(8,1,-2)

Lab6

1. Proszę napisać funkcję, która pozwoli na wypisanie: ***n*** początkowych wierszy pliku, ***n*** końcowych wierszy pliku, co ***n***-tego wiersza pliku, ***n***-tego słowa ze wszystkich wierszy i ***n***-tego znaku ze wszystkich wierszy. Nazwę pliku oraz ***n*** przekazujemy jako parametr do funkcji. Każdy podpunkt==jedna linia kodu (1.5p).

2. Wszystkie pliki z rozszerzeniem ***in*** w katalogu bieżącym traktujemy jako wyniki kolejnych serii pomiarów tych samych wielkości. Zakładamy, że w każdym z plików mamy dwie kolumny liczb (x, y). Na wyjściu chcemy otrzymać jeden plik z trzema kolumnami:

   - średnia x z danego wiersza ze wszystkich plików (numpy.average),
   - średnia y z danego wiersza ze wszystkich plików,
   - odchylenie standardowe y z danego wiersza ze wszystkich plików (numpy.std)

3. PLIKI TESTOWE: pliki.zip
   (2p).

4. Proszę napisać funkcję, tworzącą plik z instrukcjami pozwalającymi na wygenerowanie wykresu plików j.w. + wynikowego (łącznie z odchyleniem standardowym)*patrz niżej, proszę skorzystać z potrójnego cudzysłowu (1.5p).

5. Proszę sporządzić histogram słów (należy uwzględnić białe znaki, przecinki, dwukropki, nawiasy, etc.) ze wszystkich plików z określonym rozszerzeniem np. ***py*** w katalogu bieżącym (2.5p).

6. Proszę napisać funkcję, która zakładając, że w pliku znajdują się tylko wartości liczbowe, a w poszczególnych wierszach liczba kolumn może być różna, tworzymy dwie listy: do pierwszej trafiają wartości z pierwszej kolumny pod warunkiem, ze wszystkie wartości w danym wierszu konwertują się do wartości całkowitych, do drugiej natomiast wartości z pozostałych kolumn niezależnie od ich typu i liczby (2.5p).
   PLIK TESTOWy: zad5.in

* ***Matplotlib*** jest biblioteką do tworzenia wykresów (https://matplotlib.org/). Wykorzystamy ją do wygenerowania prostego wykresu. Poniżej minimum konieczne, aby ten cel osiągnąć:

```
import matplotlib.pyplot as plt

#wyrysowanie krzywej y(x), 'o' oznacza styl punktu

plt.plot(x, y, 'o')

#wyrysowanie krzywej y(x) wraz z niepewnościami

plt.errorbar(x, y, marker='*', yerr=dy)

#opis osi

plt.xlabel('x')

#zapis do pliku, format określony przez rozszerzenie w nazwie

plt.savefig('res.pdf')
```

A to może się przydać do łatwego wczytywania plików (ale dzisiaj można z tego skorzystać tylko w skrypcie generującym wykresy)

```
import numpy

x,y=numpy.loadtxt(nazwa, unpack=True)
```

Lab7

Proszę utworzyć moduł a w nim:

1. funkcję tworzącą trójkąt Pascala, do rzędu ***n****pdf (2.5p).

2. funkcję tworzącą trójkąt Eulera, do rzędu ***n****pdf (2.5p).

3. funkcję kodującą plik szyfrem Cezara*plik testowy (tekstPL.txt), przydatny może być moduł ***string*** (2p)
   Szyfr Cezara jest rodzajem szyfru podstawieniowego, w którym każda litera tekstu jawnego (niezaszyfrowanego) zastępowana jest inną, oddaloną od niej o stałą liczbę pozycji w alfabecie, literą, przy czym kierunek zamiany musi być zachowany.
4. funkcję dekodującą plik zakodowany szyfrem Cezara, wynik proszę zapisać do pliku (1p)

5. funkcję dekodującą plik z wykorzystaniem tabeli częstości*plik, na podstawie zaszyfrowanego pliku należy utworzyć tabelę częstości liter (bez rozróżniania ich wielkości) i poprzez porównanie jej z tabelą częstości występowania liter w języku polskim (***czestosci.txt***) można spróbować plik odkodować, wynik proszę zapisać do pliku (2p)

Lab8

1. Proszę napisać funkcję sprawdzającą czy elementy listy tworzą trójkę ($a_2+b_2=c_2$)/czwórkę($a_2+b_2+c_2=d_2$) pitagorejską (funkcja ma działać dla dowolnej długości "podciągu"!). Proszę zgłosić wyjątek w przypadku niepoprawnej długości listy oraz w przypadku, gdy lista nie zawiera żadnych trójek/czwórek pitagorejskich. Dla każdej trójki/czwórki proszę sprawdzić ile jest w niej wartości parzystych i nieparzystych (3p).
   Listy testowe:
   l=(1,2,2,3,2,3,6,7,1,4,8,9,4,4,7,9,2,6,9,13,6,6,7,11,3,4,12,13,2,5,14,15,2,10,11,15,1,12,12,17,8,9,12,17,1,6,18,19,6,6,17,19,6,10,15,21,4,5,20,21,4,8,19,21,4,13,16,21,8,11,16,21,3,6,22,23,3,13,18,23,6,13,18,23,9,14,20,25,12,15,16,25,2,7,26,27,2,10,25,27,2,14,23,27,7,14,22,27,10,10,23,27,3,16,24,29,11,12,24,29,12,16,21,29,2)
   l=(1,2,2,3,2,3,6,7,1,4,8,9,4,4,7,9,2,6,9,13,6,6,7,11,3,4,12,13,2,5,14,15,2,10,11,15,1,12,12,17,8,9,12,17,1,6,18,19,6,6,17,19,6,10,15,21,4,5,20,21,4,8,19,21,4,13,16,21,8,11,16,21,3,6,22,23,3,13,18,23,6,13,18,23,9,14,20,25,12,15,16,25,2,7,26,27,2,10,25,27,2,14,23,27,7,14,22,27,10,10,23,27,3,16,24,29,11,12,24,29,12,16,21,29)
   l=(3,4,5,5,12,13,7,24,25,9,40,41,6,8,10,60,80,100,18,24,30,15,8,17)
   l=(1,2,3,4,5,6,7,8,9)

2. Proszę napisać funkcję zwracającą rozstęp oraz wartość środkową parametrów przekazanych do funkcji (n wartości, lista, krotka). Na początku należy sprawdzić czy wszystkie parametry są liczbami i czy są posortowane, jeżeli nie - proszę zgłosić odpowiedni wyjątek (2p).

3. Proszę napisać funkcję obliczającą wartość całki metodą Simpsona (funkcję podcałkową, granice całkowania oraz liczbę przedziałów proszę przekazać jako parametr funkcji). Jeżeli parametry nie są poprawne proszę zgłosić odpowiedni wyjątek (2.5p).
   $h=(x_k-x_p)/(2n)$
   $s=(h/3)[f_0+4(f_1+f_3+...+f_{2n-1})+2(f_2+f_4+...+f_{2n-2})+f_{2n}]$

4. Proszę napisać funkcję znajdującą miejsce zerowe funkcji w określonym przedziale z zadaną dokładnością (np. 10^{-7}) metodą bisekcji. Jeżeli funkcja nie ma miejsca zerowego w zadanym przedziale lub jest nieokreślona w jakimś punkcie proszę zgłosić wyjątek (2.5p).
   Przykładowe funkcje:
   - x+1 [-2, 0]
   - x+1 [1, 2]
   - (x-2)*(x-2)/(x-1)-2 [0, 2]
   - (x-2)*(x-2)/(x-1)-2 [4, 6]

Lab9

1. Proszę utworzyć klasę IFS (***Iterated Function System***), a w niej (4p).:
   - metodę inicjalizującą przyjmującą jako parametr współczynniki przekształcenia oraz prawdopodobieństwa i określającą początkowe współrzędne punktu jako (0,0),
   - metodę dokonującą przekształcenia; jako parametr proszę przekazać liczbę iteracji. W każdej iteracji przy obliczaniu nowych współrzędnych punktu należy wylosować z określonym prawdopodobieństwem inną szóstkę z danego zestawu współczynników.
     Współrzędne obliczamy wg wzorów:
     $x(t+1)=a*x(t)+b*y(t)+c$
     $y(t+1)=d*x(t)+e*y(t)+f$

- metodę rysującą otrzymany wynik

2. Przykładowe zestawy współczynników wraz z prawdopodobieństwem:
    - ((0.787879, -0.424242, 1.758647, 0.242424, 0.859848, 1.408065), (-0.121212, 0.257576, -6.721654, 0.151515, 0.05303, 1.377236), (0.181818, -0.136364, 6.086107, 0.090909, 0.181818, 1.568035)), (0.90, 0.05, 0.05)

    - ((0, 0.053, -7.083, -0.429, 0, 5.43), (0.143, 0, -5.619, 0, -0.053, 8.513), (0.143, 0, -5.619, 0, 0.083, 2.057), (0, 0.053, -3.952, 0.429, 0, 5.43), (0.119, 0, -2.555, 0, 0.053, 4.536), (-0.0123806, -0.0649723, -1.226, 0.423819, 0.00189797, 5.235), (0.0852291, 0.0506328, -0.421, 0.420449, 0.0156626, 4.569), (0.104432, 0.00529117, 0.976, 0.0570516, 0.0527352, 8.113), (-0.00814186, -0.0417935, 1.934, 0.423922, 0.00415972, 5.37), (0.093, 0, 0.861, 0, 0.053, 4.536), (0, 0.053, 2.447, -0.429, 0, 5.43), (0.119, 0, 3.363, 0, -0.053, 8.513), (0.119, 0, 3.363, 0, 0.053, 1.487), (0, 0.053, 3.972, 0.429, 0, 4.569), (0.123998, -0.00183957, 6.275, 0.000691208, 0.0629731, 7.716), (0, 0.053, 5.215, 0.167, 0, 6.483), (0.071, 0, 6.279, 0, 0.053, 5.298), (0, -0.053, 6.805, -0.238, 0, 3.714), (-0.121, 0, 5.941, 0, 0.053, 1.487)), prawdopodobieństwo jednakowe

    - ((0.824074, 0.281428, -1.88229, -0.212346, 0.864198, -0.110607), (0.088272, 0.520988, 0.78536, -0.463889, -0.377778, 8.095795)), (0.8, 0.2)

3. Proszę utworzyć klasę Wektor, a w niej metody przeciążające operatory dodawania, odejmowania, mnożenia (mnożenie wektora przez liczbę) oraz metodę str. Proszę napisać także metody zwracające odpowiednio długość wektora, iloczyn skalarny, wektorowy i mieszany (proszę wykorzystać wcześniej zdefiniowane metody) (4p).

4. Proszę utworzyć klasę z metodami statycznymi obliczającymi odpowiednio (2p):

    - strumień indukcji magnetycznej: $\Phi=B \bullet S$
    - siłę Lorentza $F=q(E+v \times B)$
    - pracę siły Lorentza $W=qE \bullet v$

5. Metoda statyczna (wywoływana przez klasę, bez parametru odnoszącego się do instancji):
   @staticmethod
   def metoda():
   ....

Lab10

1. Proszę napisać iterator zwracający kolejne wyrazy ciągu Fibonacciego dwoma sposobami (jedna lub dwie klasy) i porównać ich wykorzystanie (2p).

2. Celem programu jest implementacja tzw. ***automatu komórkowego***. Składa się on z N komórek, każda w stanie 0 lub 1. W kolejnych iteracjach stan danej komórki zmienia się zgodnie z określoną regułą, na podstawie stanu danej komórki oraz jej dwóch najbliższych sąsiadów (prawego i lewego) w poprzedniej iteracji. Regułę zapisuje się jako binarną reprezentację liczby zapisanej w systemie dziesiętnym, np. dla reguły nr 30 jej reprezentacja binarna to 00011110. Jeżeli każda komórka może być w dwóch stanach (0 lub 1), to możliwych jest osiem kombinacji stanu komórki i jej dwóch najbliższych sąsiadów.

   Możemy utworzyć tabelę, w górnym wierszu której przedstawiony jest stan danej komórki i jej dwóch najbliższych sąsiadów (lewy sąsiad, komórka, prawy sąsiad) w danej chwili czasowej, a w wierszu dolnym wpisujemy regułę automatu, która określa stan komórki centralnej w chwili kolejnej (przyjmujemy periodyczne warunki brzegowe - lewym sąsiadem komórki o indeksie ***0*** jest komórka o indeksie ***N-1***, prawym sąsiadem komórki o indeksie ***N-1*** jest komórka o indeksie ***0***):

111	110	101	100	011	010	001	000
0	0	0	1	1	1	1	0

Przykład:
t=0: 0000000001000000000
t=1: 0000000011100000000
t=2: 0000000110010000000
t=3: 0000001101111000000

Proszę utworzyć klasę Automat, a w niej:

- metodę inicjalizującą przyjmującą dwa parametry całkowite i jeden logiczny, pierwszy parametr określa liczbę komórek, drugi regułę automatu, a trzeci sposób inicjalizacji stanu początkowego: losowy bądź zera z wyjątkiem komórki w środku, równej 1
- metodę ewolucji automatu z parametrem określającym liczbę iteracji
- metodę wypisującą na ekran przebieg ewolucji (zamiast jedynek proszę wyświetlać *, a zamiast zer - spacje)

  Proszę uruchomić program dla automatu o długości 31, dla reguł 90, 94 i 182 oraz 16 iteracji (5p)

3.

4. Proszę utworzyć iterator zwracający liczbę pseudolosową z przedziału [0,1) generowaną wg wzoru $X_{n+1} = (aX_n + c)\%m$, dla $m = 2^{48}$, $a = 44485709377909$, $c = 0$, $x_0 = 1$. Korzystając z zaimplementowanego iteratora proszę sprawdzić ile punktów trzeba wylosować, aby obliczyć wartość liczby ***pi*** z zadaną dokładnością, np. 10^{-7} (stosujemy metodę Monte Carlo) (3p)

   Koło o promieniu 1 wpisujemy w kwadrat o boku 2 i umieszczamy ich środki w początku układu współrzędnych. Stosunek pól tych figur jest równy stosunkowi liczby trafień w ich obszar, przy losowaniu dużej liczby punktów wewnątrz kwadratu .

Lab11

1. Proszę napisać klasę ***Calka*** z metodą inicjalizacyjną określającą granice całkowania, liczbę kroków oraz funkcję podcałkową (proszę skontrolować poprawność przekazanych parametrów) oraz metodą abstrakcyjną obliczającą wartość całki.
   Następnie proszę utworzyć klasy dziedziczące po klasie ***Calka*** z metodami obliczającymi wartość całki odpowiednio metodą trapezów lub Simpsona, w metodzie proszę umieścić komentarz dokumentacyjny. Potrzebne wzory są w

pliku: calki1.pdf (3.(3)p)

2. Proszę napisać klasę implementującą stos, klasa ma obsługiwać możliwość tworzenia pustego stosu bądź inicjalizacji istniejącym stosem (obiektem klasy), dodawania i usuwania elementu, dodawania elementów innego stosu, zwracania rozmiaru i wypisywania stosu.
   Następnie proszę napisać klasę dziedziczącą po klasie stosu i implementującą stos posortowany (rosnąco lub malejąco). W tym przypadku element/elementy innego stosu można do stosu dodać pod warunkiem zachowania porządku sortowania.
   Proszę sprawdzić jaki jest średni rozmiar posortowanego stosu, który wypełniamy całkowitymi liczbami losowymi z przedziału [0,100] losując 100 wartości (średnia po 100 powtórzeniach) (3.(3)p)

3. Proszę zaimplementować klasę pozwalającą na zliczanie linii, słów i znaków w pliku (metody inicjalizująca i zliczająca). W klasie proszę także zaimplementować metodę statyczną zwracają komunikat analogiczny do komunikatu zwracanego przez polecenie systemowe linuxa ***wc*** w przypadku jednoczesnego zliczania dla kilku plików (3.(3)p)
   Przykład:
   $wc AA.py BB.py
   50 91 944 AA.py
   80 117 1281 BB.py
   130 208 2225 razem

Lab12

Uwaga: przy przesyłaniu zadania wszystkie potrzebne pliki proszę spakować

1. Proszę zapisać plik rozszerzenia (mod.c) oraz skrypt instalacyjny (setup.py). Proszę tak zmodyfikować plik rozszerzenia, aby otrzymywany wynik był poprawny, tj. np. :
   met(1,2) #3
   met(1,2,5) #8
   met(1,2,5,[2,3,4]) #17
   (2p).

2. Proszę zaimplementować w Pythonie i w C funkcję losującą współrzędne N punktów w kwadracie o boku 1. Proszę zliczyć jaki procent punktów trafia odpowiednio w kwadraty o boku [0.1, 0.2, ..., 1]. Funkcję napisaną w C proszę wywołać z programu w Pythonie. Proszę porównać wyniki oraz czasy wykonania obu funkcji w zależności od liczby punktów [10^2, 10^3, ..., 10^6].(4p).

3. Proszę zmodyfikować plik rozszerzenia tak, aby była w nim funkcja, do której jako parametr będziemy przekazywać z Pythona słownik (klucze i wartości - liczby

losowe z przedziału [10,100]). Dla każdej pary (klucz,wartość) proszę wywołać napisaną w języku C funkcję zwracającą największy wspólny dzielnik (poniżej) dwóch liczb przekazanych jako parametr, a wynik proszę zapisać w słowniku, który proszę zwrócić do Pythona (4p).

AlgorytmEuklidesa(a, b)
1: if b = 0 then
2: return a
3: else
4: return AlgorytmEuklidesa(b, a mod b)
5: end if

Lab13

1. Proszę utworzyć klasę definiującą współrzędne punktu na płaszczyźnie. Obie współrzędne proszę zdefiniować jako własności (metoda inicjalizacyjna bezparametrowa) (1p)

2. Proszę zdefiniować funkcje dodawania i odejmowania współrzędnych (z wykorzystaniem wcześniej zdefiniowanej klasy) oraz opatrzyć je dekoratorem (implementowanym jako funkcja) sprawdzającym czy współrzędne leżą w określonym zakresie, jeżeli nie - proszę zgłosić wyjątek (3p)

3. Proszę utworzyć klasę z metodami statycznymi obliczającymi obwód i pole trójkąta lub czworokąta (dających się wpisać w okrąg, odpowiednio wzory Herona i Brahmagupty), zdefiniowanych poprzez współrzędne wierzchołków (klasa z zadania 1) (3p)

   Wzór Herona: $P=[p(p-a)(p-b)(p-c)]^{1/2}$, gdzie: a,b,c - długości boków, p - połowa obwodu
   Wzór Brahmagupty: $P=[(p-a)(p-b)(p-c)(p-d)]^{1/2}$, oznaczenia j.w.

4. Proszę utworzyć dekorator (implementowany jako klasa) umożliwiający zliczenie liczby wywołań funkcji, z metodą statyczną zwracającą wynik (3p)

Lab14

1. Proszę zapisać wersję 0.0 (plik ***kal.py***) kalkulatora i sprawdzić, że działa.
2. Proszę rozbudować kalkulator: wszystkie cyfry, działania: +, -, *, /, %, **, (,), =. (1p).

3. Proszę dodać możliwość rysowania wykresu funkcji korzystając z modułu ***matplotlib.pyplot***; użytkownik ma mieć możliwość określenia funkcji oraz dziedziny (2p).
4. Proszę dodać możliwość całkowania metodą ***quad*** z modułu ***scipy.integrate***; użytkownik ma mieć możliwość określenia funkcji oraz granic całkowania (2p).
5. Proszę dodać możliwość znajdowania miejsc zerowych podanej funkcji metodą ***bisect*** z modułu ***scipy.optimize***; użytkownik określa funkcję oraz przedział poszukiwania (2p).
6. Proszę dodać możliwość fitowania danych (dane testowe plik ***dane.dat***) wczytywanych z pliku (np. ***numpy.loadtxt***), którego nazwę użytkownik podaje, z wykorzystaniem funkcji ***leastsq*** z modułu ***scipy.optimize***. Proszę narysować wykres obrazujący dane oraz dofitowaną funkcję z wykorzystaniem modułu ***matplotlib.pyplot*** (3p).