Laboratorium 1-2

Zadanie 1: Współczynnik BMI

Napisz program, który na podstawie danych o wzroście (podawanym w cm) i wadze (podawanej w kg) otrzymanych od użytkownika wypisze jego wskaźnik BMI (*Body Mass Index*) oraz jego słowną interpretację. Sprawdź (np. w internecie) w jaki sposób oblicza się taki wskaźnik, a następnie spróbuj przenieść ten wzór do języka Python.

Zadanie 2: Równianie kwadratowe

Proszę napisać program rozwiązujący równanie kwadratowe w postaci:
a*x^2 + b*x + c = 0.

Rozwiązanie:

```
from math import sqrt

a=float(input("a="))
b=float(input("b="))
c=float(input("c="))

d=b*b-4*a*c
if d>=0:
    p=sqrt(d)
    x1=(-b-p)/(2*a)
    x2=(-b+p)/(2*a)
    print("x1=%0.4f" % x1)            # stary sposób (niezalecany)
    print("x2={:0.4f}".format(x2))    # nowy sposób
else:
    print("brak pierwiastków")
# end if
```

Proszę zwrócić uwagę na:
- Błędy składniowe sprawdzane przed rozpoczęciem wykonania program,
- Dostępność do funkcji sqrt sprawdzana w czasie wykonania,
- Różne sposoby importu funkcji z modułów biblitecznych,
- Różne sposoby formatowania wyników.

Jeżeli chcemy zabezpieczyć program przed wprowadzaniem niepoprawnych danych, można skorzystać z mechanizmu wyjątków, czyli:

```
. . .
while True:
    try:
        a=float(input("a="))
        b=float(input("b="))
        c=float(input("c="))
        break
    except:
        print("niepoprawne dane")
    # end try
# end while
. . .
```

Zadanie 3: Za dużo, za mało

Napisz grę, w której komputer losuje liczbę z zakresu <1,100>, a użytkownik musi ją odgadnąć. Jeśli użytkownik wpisze za dużą liczbę, program powinien podpowiedzieć, że jest zbyt duża. Jeśli zbyt małą - analogicznie. Podpowiedzi powinny naprowadzać użytkownika dopóki nie odgadnie właściwej liczby.

Wskazówka: zapoznaj się z funkcjami zawartymi w bibliotece random.

Zadanie 3a:

Proszę zmodyfikować program z zadania 3, tak aby komputer grał sam ze sobą.

Zadanie 4. Urodziny

Sprawdź, jakie jest prawdopodobieństwo, że w grupie 40 osób 2 osoby urodziły się tego samego dnia roku. Następnie pokaż, w jaki sposób zmienia się to prawdopodobieństwo wraz ze wzrostem liczby osób (wypisując wyniki dla kolejnych grup od 20 do 50 osób).

Wskazówka: Wyznaczając prawdopodobieństwo można wykonać symulację dla reprezentatywnie dużej liczby przypadków, np. powtarzając losowanie grupy osób 10000 razy.

Rozwiązanie (Pojedynczy eksperyment):

```
import random

t = {}
for i in range(1, 366): t[i] = 0
for i in range(n):
   dzien = random.randint(1, 365)
   t[dzien] = t[dzien]+1
# end for
para = False
for i in range(1, 366):
  if t[i]>1: para = True
# end for
print(para)
```

Rozwiązanie (Cała symulacja):

```
import random

t = {}
for n in range(20, 51):
  licz = 0
  for j in range(10000):
    for i in range(1, 366): t[i] = 0
    for i in range(n):
      dzien = random.randint(1, 365)
      t[dzien] = t[dzien]+1
    # end for
    para = False
    for i in range(1, 366):
      if t[i]>1: para = True
    # end for
    if para: licz = licz+1
  # end for
  print("%d  %.1f" % (n, licz/100.0))
# end for
```

Zadanie 5: Hipoteza Collatza

Zaczynając od dowolnej liczby naturalnej przekształcamy ją według zasady:

- gdy jest parzysta dzielimy ją przez 2
- gdy jest nieparzysta, mnożymy ją przez 3 i dodajemy 1

Hipoteza mówi, że nigdy taki proces nie wpadnie w cykl i zawsze zakończymy na liczbie 1.

1. Proszę napisać program weryfikujący hipotezę Collatza dla liczb mniejszych od miliona.
2. Proszę należć liczbę dla której trzeba wykonać dokładnie 100 przekształceń
3. Proszę znaleźć liczby dla których liczba przekształceń jest równa liczbie startowej.

Zadanie 6: Liczby doskonałe

Napisz program, który znajdzie i wypisze wszystkie liczby doskonałe z mniejsze od 10^5.

Wskazówka: liczba doskonała to taka, której wartość jest równa sumie jej dzielników właściwych (mniejszych od niej samej). Proszę napisać funkcję, która dla danej liczby zwraca sumę jel podzielników właściwych (mniejszych niż ta liczba).

Przygotuj 2 warianty programu - naiwny, w którym sprawdzane będą wszystkie podzielniki właściwe danej liczby oraz zoptymalizowany, w którym sprawdzanie kończy się na pierwiastku kwadratowym z sprawdzanej liczby.

Wariant naiwny i bardziej efektywny:

```
from math import sqrt, floor

def sp(n):
    s = 0
    p = 1
    while p<n:
        if n%p==0: s = s+p
        p = p+1
    # end while
    return s
# end def

def sp2(n):
    if n==1: return 0
    s = 1
    p = 2
    g = floor(sqrt(n))
    while p<g:
        if n%p==0: s = s+p+n//p
        p = p+1
    # end while
    if p*p==n: s = s+p
    return s
# end def
```

Utwórz własny moduł zawierający funkcje sp i sp2.
Porównaj czasy wykonania programów z użyciem funkcji sp i sp2.

Rozwiązanie:

Liczby doskonałe:

```
from mylib import sp, sp2

for i in range(2, 100000):
  if sp(i)==i: print(i, j)
# end while
```

Zadanie 6a:

Dwie liczby nazywamy zaprzyjaźnionymi jeżeli suma podzielników właściwych jednej liczby jest równa drugiej liczbie. Rozszerz program z poprzedniego zadanie, tak by w podanym zakresie znajdował wszystkie liczby zaprzyjaźnione

Rozwiązanie (naiwne):

```
from mylib import sp, sp2

for i in range(2, 100000):
  for j in range(2, 100000):
    if sp(i)==sp(j) and i<j: print(i, j)
  # end for
# end for
```

Rozwiązanie (lepsze):

```
from mylib import sp, sp2

for i in range(2, 100000):
  j = sp2(i)
  if sp2(j)==i and i<j: print(i, j)
# end for
```

Zadanie 7: Kalendarz

Napisać funkcję, która na podstawie daty wyznacza numer dnia tygodnia.

```
def dzien(r,m,d):
  ''' Funkcja zwraca numer dnia tygodnia dla dowolnej daty
      w latach 1900-2099. Niedziela ma numer 0. '''

  rm = (-1,0,3,3,6,1,4,6,2,5,0,3,5)
  dt = d + rm[m] + (r-1900) + (r-1900)//4
  if m<3 and (r%4==0 and (r%100!=0 or r%400==0)): dt=dt-1
  return dt%7
# end def
```

Wykorzystując powyższą funkcję wyznaczyć:

1. Najbliższe 10 piątków wypadających 13 dnia miesiąca,
2. Najbliższe 10 miesięcy, które będą miały po 5 niedziel.
3. Zmodyfikuj funkcję aby działała poprawnie po roku 2099.

Zadanie 8: Master Mind

Zrealizuj w Pythonie grę Master Mind. Zamiast kolorów możemy użyć cyfr z zakresu <1,6> lub liter A..F. Użytkownik będzie w tej grze *Odgadującym*, a program *Kodującym* (generującym losowy kod złożony z 4 znaków).

Rozwiązanie:

Funkcje pomocnicze:

```
import random

def gen():
    ”Generuje 4 znakowy napis złożony z cyfr 1-6”
    res = ""
    for i in range(4):
        res = res+str(random.randint(1, 6))
    return res
# end def

def spr(kod, proba):
    ”Zwraca liczbę cyfr na swoich miejscach I na nieswoich miejscach
    W postaci krotki (lc,lb)”
    k = list(kod)
    p = list(proba)
    lc = 0
    for i in range(4):
        if p[i] == k[i]:
            lc = lc + 1
            p[i] = -1
            k[i] = -2
        # end if
    # end for
    lb = 0
    for i in range(4):
        for j in range(4):
            if p[i] == k[j]:
                lb = lb + 1
                p[i] = -1
                k[j] = -2
        # end if
    # end for
    return (lc, lb)
# end def
```

Główny program

```
kod = gen()
koniec = False
while not koniec:
    proba = input(">>")
    p = spr(kod, proba)
    print(10*" ",p)
    koniec = (p==(4, 0))
# end while
print("Gratuluje")
```

Zmodyfikuj program aby liczył liczbę ruchów i w zależności od ich liczby wypisywał stosowny komunikat o graczu.