Probleme cu algoritmi elementari

Problema propusă #1

Se citește un număr natural `n`. Afișați suma primelor `n` numere naturale nenule și suma primelor `n` numere naturale nenule **pare**.

Indicație: puteți rezolva problema atât printr-o structură repetitivă cât și prin utilizarea unei formule matematice.

```
n = int(input("n ="))
suma = 0
suma_para = 0
for i in range(1, n + 1):
    suma += i
for i in range(1, n + 1):
    suma_para += 2 * i
print("Suma primelor", n, "numere naturale nenule:", suma)
print("Suma primelor", n, "numere naturale nenule pare: ", suma_para)

#  Sau, folosind formula
# n = int(input("n ="))
# suma = n * (n + 1) / 2
# suma_para = suma * 2
# print("Suma primelor", n, "numere naturale nenule:", suma)
# print("Suma primelor", n, "numere naturale nenule pare: ", suma_para)
```

Problema propusă #2

Se citesc de la tastatură două numere naturale nenule, `a` și `b`, fiecare pe o linie. Să se afișeze cel mai mare divizor comun al celor două numere.

Indicație: Algoritmul lui Euclid.

```
a = int(input())
b = int(input())
if a < b:
    a, b = b, a
while b > 0:  #algoritmul este ușor optimizat, pe internet cu siguranță se găsește o va
rianta mai simplă
    a = a % b
    a, b = b, a
print("Cel mai mare divizor comun al celor două numere:", a)
```

Problema propusă #3

Se citește un număr natural `n` >= 2. Determinați dacă acesta este prim sau nu și afișați un mesaj corespunzător pentru ambele cazuri.

Indicație: oportunitatea perfectă pentru a folosi o structură repetitivă de tip 'for... else...'

```
n = int(input())
for i in range(2, n):  #o optimizare pentru acest algoritm va fi menționată
    if n % i == 0:
        print("Nu este prim!")
        break
else:
    print("Este prim!")
```

Problema propusă #4

Se citesc de la tastatură două numere naturale nenule, `a` și `b`, și un operator din lista [+, -, *, /, **], fiecare pe o linie. Să se afișeze `a op b = rez`, unde `rez` reprezintă rezultatul operației descrise.

Indicație: Deși metoda de rezolvare prin folosirea unor structuri condiționate de tip 'if... elif... else...' nu ar fi greșită, structura 'match' este recomandată. Cazul default nu trebuie ignorat

```
x = int(input())
y = int(input())
op = input()
match op:
    case "+":
        print(f"{x} {op} {y} = {x + y}")
    case "-":
        print(f"{x} {op} {y} = {x - y}")
    case "*":
        print(f"{x} {op} {y} = {x * y}")
    case "/":
        print(f"{x} {op} {y} = {x / y}")
    case "**":
        print(f"{x} {op} {y} = {x ** y}")
    case _:
        print("Acesta nu este un operator!")
```

Problema propusă #5

Se citesc de la tastatură trei numere naturale nenule, fiecare pe o linie. Decideți dacă acestea pot reprezenta o oră, respectiv primul număr să reprezinte ora, al doilea minutul și ultimul secunda.

Indicație: Expresiile '0 < n and 10 > n' și '0 < n < 10' sunt echivalente și sunt acceptate în limbaj python

```
a = int(input())
b = int(input())
c = int(input())
if 0 <= a < 24 and 0 <= b < 60 and 0 <= c < 60:
```

```
    print("Da, aceasta este o oră.")
else:
    print("Nu, aceasta nu este o oră")
```

Problema propusă #6

Se citesc de la tastatură coeficienții reali ai unei ecuații de gradul al doilea, fiecare pe o linie (primul fiind nenul). Determinați coordonatele punctului de extrem asociat graficului funcției și natura acestuia.

Indicație: x = - b / 2a, y = f(x).

```
a = float(input())
b = float(input())
c = float(input())
d = (b ** 2 - 4 * a * c) ** 1 / 2  #sau, cu import math, math.sqrt
print(-b / (2 * a), -d / (4 * a))
```

Problema propusă #7

Se dă un număr natural nenul `n` > 1. Afișați descompunere în factori primi a numărului `n` astfel:

nr.1_prim puterea_la_care_apare

nr.2_prim puterea_la_care_apare

ș.a.m.d.

Indicație: întreabă un tutore pentru sfaturi dacă ești blocat

```
n = int(input())
d = 2
while d <= n:
    p = 0
    while n % d == 0:
        p += 1
        n = n / d
    if p != 0:
        print(d, p)
    d += 1
```

Problema propusă #8

Se citește un număr natural nenul `n`. Să se afișeze primele n^2 numere naturale sub forma unui pătrat.

Exemplu: pentru n = 2 avem

1 2

3 4

Indicație: configurați funcția de afișare print()

```
n = int(input())
for i in range(n):
    for j in range(n):
        print(i * n + j + 1, end=" ")
    print()
```

Problema propusă #9

Se citește un număr natural nenul `n`. Să se afișeze primele `n` linii ale triunghiului lui Pascal.

Exemplu: pentru n = 3 avem

1

1 1

1 2 1

Indicație: Elementul k de pe linia i este dat de formula $C_k^i = \frac{i!}{k!(i-k)!}$

```
n = int(input())
for i in range(n):
    p = 1
    print(p, end=" ")
    for j in range(i):
        p *= (i - j) / (j + 1)  #work smarter, not harder
        print(int(p), end=" ")
    print()
```

Probleme cu liste #1

Scrieți un program care să efectueze suma și produsul tuturor elementelor dintr-o listă.

```
l = [1, 4, 6, 7, 5, 4]
print(sum(l))
suma = 0
produs = 1
for i in l:
    suma += i
    i += 1
for i in l:
    produs *= i
print(suma)
print(produs)
```

Probleme cu liste #2

Scrieți un program care să determine minimul dintr-o listă.

```
l = [4, 6, 8, 9, 2]
print(min(l))
minim = l[0]  #initializez minimul cu o valoare din lista care urmeaza sa fie comparata
 cu toate elementele listei
for i in l:
    if minim > i:
        minim = i
print(minim)
```

Probleme cu liste #3

Care atribuiri sunt incorecte? Ce fac cele corecte pentru lista `ls=list(range(7))`?

```
ls[4:5] = 45

ls[4:5] = [45]

ls[4:5] = "ab"

ls[4:5] = ['ab']

ls[4:5] = (32,) # (32,) este un tuplu cu un singur element

ls[4:5] = [100, 102]

ls[4:5] = [[100, 102], 88]

ls[4:5] = []

ls[4:5] = ls[5:4]

ls[4:5] = ls[3]

ls[4:5] = ls[3:]

ls[4:5] = [ls[3]]

ls[4:4] = ['ab']

ls[4:4] = 34

ls[4:4] = []
```

```
ls = list(range(7))  # ls = [0, 1, 2, 3, 4, 5, 6]

# ls[4] = 45 # inlocuieste elementul de pe pozitia 4 cu 45

# ls[4:5] = 45 # eroare (45 nu e iterabil (lista, string, tuplu))
```

```
# ls[4:5] = [45] # inlocuieste elementul de pe pozitia 4 cu 45

# ls[4:5] = "ab" # inlocuieste elementul de pe pozitia 4 cu 'a', 'b' (lungimea listei c
reste cu 1)
# ls = [0, 1, 2, 3, 'a', 'b', 5, 6]

# ls[4:5] = ['ab'] # inlocuieste elementul de pe pozitia 4 cu 'ab'
# ls = [0, 1, 2, 3, 'ab', 5, 6]

# ls[4:5] = (32,) # inlocuieste elementul de pe pozitia 4 cu 32

# ls[4:5] = [100, 102] #ninlocuieste elementul de pe pozitia 4 cu 100, 200 (lungimea li
stei creste cu 1)
# ls = [0, 1, 2, 3, 100, 200, 5, 6]

# ls[4:5] = [[100, 102], 88] # inlocuieste elementul de pe pozitia 4 cu [100, 200], 88
# lungimea listei creste cu 1
# ls = [0, 1, 2, 3, [100, 200], 88, 5, 6]

# ls[4:5] = [] # stergem elementul de pe pozitia 4 (lungimea scade cu 1)

# ls[4:5] = ls[5:4] # ls[5:4] = [], deci stergem elementul de pe pozitia 4 (lungimea sc
ade cu 1)

# ls[4:5] = ls[3] # eroare (ls[3] nu e iterabil (lista, string, tuplu))

# ls[4:5] = ls[3:] # il inlocuieste pe 4 cu elementele din lista l[3:] = [3, 4, 5, 6]
# ls = [0, 1, 2, 3, 3, 4, 5, 6, 5, 6]

# ls[4:5] = [ls[3]] # il inlocuieste pe 4 cu 3


# ls[4:4] = ['ab'] # adauga pe pozitia 4 (intre 3 si 4) elementul 'ab' (lungimea creste
 cu 1)
# ls = [0, 1, 2, 3, 'ab', 4, 5, 6]

# ls[4:4] = 34 # eroare (34 nu e lista)

# ls[4:4] = [] # ramane la fel

# ls[4:4] = [[]] # adauga [] intre 3 si 4 (lungimea creste cu 1)
# ls = [0, 1, 2, 3, [], 4, 5, 6]

print(ls)
```

Probleme cu liste #4

Se dă șirul de caractere `s = "acest exemplu"`. Ce se va afișa pentru felierile:

```
s[0:4],

s[0:4:2],

s[0:4:-2],

s[0::2],

s[::100],

s[::-100],

s[-3:-1],

s[-1:4],

s[-12:4],

s[len(s):]
```

```
s = "acest exemplu"
print(f"s == 'acest exemplu'")
print("s[0:4] -", s[0:4])

print("s[0:4:2] -", s[0:4:2])

print("s[0:4:-2] -", s[0:4:-2])

print("s[0::2] -", s[0::2])

print("s[::100] -", s[::100])

print("s[::-100] -", s[::-100])

print("s[-3:-1] -", s[-3:-1])

print("s[-1:4] -", s[-1:4])

print("s[-12:4] -", s[-12:4])

print("s[len(s):] -", s[len(s):])
```

Probleme cu liste #5

Știind că `l = [1, 2, 3]`, care este outputul următorului fragment de cod, rulat în ordinea aceasta:

```
l[len(l):] = [4, 5, 6]
print(l)
l[:0] = [-3, -2, -1, 0]
print(l)
```

```python
l = [1, 2, 3]
l[len(l):] = [4, 5, 6]  # adaugam la finalul listei elementele 4, 5, 6
print(l)
l[:0] = [-3, -2, -1, 0]  # adaugam la inceputul listei elementele -3, -2, -1, 0
print(l)  # l = [-3, -2, -1, 0, 1, 2, 3, 4, 5, 6]
```

Probleme cu liste #6

Scrieți un program care primește o listă și afișează lista în urma eliminării elementelor de pe pozițiile 1, 4 și 6. Propuneți cât mai multe modalități de a șterge un element de pe o poziție.

TEORIE:

- `del ls[i]` - șterge elementul de pe poziția `i` al listei `ls`
- `ls.pop(i)` - șterge elementul de pe poziția `i` al listei `ls` și returnează valoarea elementului șters

```python
l = [1, 4, 6, 8, 5, 8, 9, 7]

if len(l) >= 6:
    del l[6], l[4], l[1]

    #     sau

    # l.pop(6), l.pop(4), l.pop(1)

    #     sau

    # l[6:7] = []
    # l[4:5] = []
    # l[1:2] = []

print(l)
```

Așa nu

Exemplu de așa nu la ex de mai sus

```python
l=[1, 4, 6, 8, 5, 8, 9, 7]
if len(l) >= 6:
    del l[1], l[4], l[6] #IndexError: list assignment index out of range

    # noi stergand elementul de pe pozitia 1 si cel de pe pozitia 4 (care o sa fie cel
 de pe pozitia 5 din lista originala),

    # sirul nostru nu mai este de lungime 8, ci de lungime 6,

    # adica ultimul element din lista o sa se afle pe pozitia 5, neavand astfel ce ele
```

```
ment sa stergem

print(l)
```

Probleme cu liste #7

Dată o listă de numere naturale și un număr `k`, propuneți mai multe modalități de a șterge primele `k` elemente din listă.

```
l = [1, 4, 6, 8, 5, 8, 9, 7]
print(len(l))
k = int(input())
del l[:k]  #daca k<len(l) o sa ne returneze elementele ramase, daca k>=len(l) ne return
eaza lista vida
print(l)
```

```
l = [1, 4, 6, 8, 5, 8, 9, 7]
k = int(input())
for i in range(k):
    del l[
        0]  # stergem de fiecare data elementul de pe prima pozitie, repetand acest pro
ces de k ori ajungand sa stergem primele k elemente
    i += 1
print(l)
```

```
l = [1, 4, 6, 8, 5, 8, 9, 7]
k = int(input())
for i in range(k):
    l.pop(0)  #sintaxa de tip l.pop(pozitie); sterge si apoi returneaza elementul de pe
 pozitia respectiva
print(l)
```

```
l = [1, 4, 6, 8, 5, 8, 9, 7]  # acest cod este gresit, dar de ce?
k = int(input())
for i in range(k):
    print(i)
    print(l[i])
    l.remove(l[i])
print(l)
```

```
l = [1, 4, 6, 8, 5, 8, 9, 7]
k = int(input())
for i in range(k):
    l.remove(l[0])  # sintaxa de tip l.remove(element) sterge prima aparitie;

print(l)
```

Probleme cu liste #8

Construiește o listă cu pătratele numerelor impare de la 1 la 10, folosind comprehensiune.

```
rez = [x ** 2 for x in range(1, 11) if x % 2 == 1]
print(rez)
# [1, 9, 25, 49, 81]
```

Probleme cu liste #9

Dată o listă de cuvinte, creează o nouă listă care conține doar prima literă din fiecare cuvânt.

```
cuvinte = ["masa", "carte", "floare", "soare"]
prime_litere = [c[0] for c in cuvinte]
print(prime_litere)
# ['m', 'c', 'f', 's']
```

Probleme cu liste #10

Scrie un program care citește un număr `n` și construiește o listă cu pătratele numerelor de la 1 la `n`, folosind `append()`.

```
n = int(input("n = "))
lista = []
for i in range(1, n + 1):
    lista.append(i ** 2)
print(lista)
```

Probleme cu liste #11

Citește o listă de numere și construiește o nouă listă care conține doar elementele pare, folosind `append()`.

```
numere = [3, 8, 5, 10, 12, 7]
pare = []
for x in numere:
    if x % 2 == 0:
        pare.append(x)
print(pare)
# [8, 10, 12]
```

Problema propusa #10

Să se scrie un program care să determine numărul de cifre pare și numărul de cifre impare ale unui număr natural citit de la tastatură.

```python
n = int(input())

nr_cif_pare = 0
nr_cif_impare = 0
while n != 0:
    uc = n % 10
    if uc % 2 == 0:
        nr_cif_pare += 1
    else:
        nr_cif_impare += 1
    n //= 10

print("nr cifre pare:", nr_cif_pare)
print("nr cifre impare:", nr_cif_impare)
```

Problema propusa #11

Se citește un număr natural `n` și o cifră `c` > 0. Să se adauge cifra `c` la începutul și la sfârșitul lui `n`.

Exemplu: pentru `n = 345` și `c = 1` se va afișa `13451`.

Met. I - numeric

```python
n = int(input("Baga nr: "))
c = int(input("Baga cifra: "))

temp = c * (10 ** len(str(n))) + n
rez = temp * 10 + c

print(rez)
```

Met. II - stringuri

```python
n = input("Introdu numar: ")
c = input("Introdu cifra: ")

n = c + n + c

print(n)
```

Problema propusa #12

Se citește un număr natural `n` și un număr prim `p`. Să se afișeze la ce putere apare `p` in descompunerea în factori primi a lui `n`.

```python
n = int(input("n = "))
p = int(input("p = "))
```

```
putere = 0
while n % p == 0:
    putere += 1
    n = n // p

print(putere)
```

Problema propusa #13 (oglindit)

Se citește un număr natural `n`. Să se afișeze oglinditul său.

Exemplu:

Input: 12345

Output: 54321

```
n = int(input("n = "))
ogl = 0

while n != 0:
    uc = n % 10
    ogl = ogl * 10 + uc
    n //= 10

print(ogl)
```