# PROIECT SDA: PROBLEMA 3

*(Condrat Mihai, Guiţa Bianca Oana, Mihalea Andreas, GRUPA 211)*

## 1. FORMULAREA CERINŢEI:

- ***CERINŢA INIŢIALĂ***: Gasiţi o succesiune de mutări a.i. introducând: **1 2 3 4 5** în stiva (în aceasta ordine) la final să se afişeze: **3 2 4 5 1**.

- ***REFORMULAREA CERINŢEI***: Se citesc 'n' numere naturale (a1, a2, …, an) care sunt adăugate într-o stiva (s = [an, a(n-1), …, a1]). Gasiţi o metodă generală (pentru oricare 'n' număr natural citit) astfel încât secvenţa obţinută să respecte patternul dat pentru n = 5 (din exemplul primit). Implementati metoda gasită în Python.

## 2. IDEEA PROBLEMEI:

Verificăm cum ar trebui sa arate secvenţa pentru n = 1, …, 7 si observăm urmatorul pattern:

```
n = 7: n(a1) n(a2) n(a3) n(a4) p p p n(a5) p n(a6) p n(a7) p p
n = 6: n(a1) n(a2) n(a3) p p n(a4) p n(a5) p n(a6) p p
n = 5: n(a1) n(a2) n(a3) p p n(a4) p n(a5) p p
n = 4: n(a1) n(a2) p n(a3) p n(a4) p p
n = 3: n(a1) n(a2) p n(a3) p p
n = 2: n(a1) n(a2) p p
n = 1: n(a1) p
```

```
(1 2 3 4 5 6 7 -> 4 3 2 5 6 7 1)
(1 2 3 4 5 6 -> 3 2 4 5 6 1)
(1 2 3 4 5 -> 3 2 4 5 1)
(1 2 3 4 -> 2 3 4 1)
(1 2 3 -> 2 3 1)
(1 2 -> 2 1)
(1 -> 1)
```

## 3. STRUCTURĂ SI FUNCŢIONALITATE:

Rezolvarea propusă de noi se bazează pe partiţionarea stivei în două jumătăţi pe care operăm în moduri diferite. Pentru prima jumatate, încărcăm primele n/2 numere şi apoi apelăm un pop. Pentru cea de-a doua jumătate, încărcăm numerele rămase până la al n-lea şi, imediat după introducerea lor în stiva, le dăm pop.

Variabilele necesare pentru înţelegerea exemplului prezentat în continuare:

- n := lungime stack
- i := iteraţia curentăîn parcurgerea stackului
- e := reţine jumătatea indicilor
- contor := partiţioneaza stackul in două parţi (pentru a controla pop-urile)

## 4. EXEMPLU REZOLVAT:

//POZA TATI

## 5. IMPLEMENTARE IN PYTHON:

Codul complet poate fi găsit pe acest cont: github.com/condratmihai.

```
from random import randint
from problem import Problem

def numar(x,stack):
     stack.append(x); #O(n)

def p(arr,stack):
     arr.append(stack.pop()); #O(n)

def div2(a):
     if a%2 == 0:
          return int(a/2-1);
     else:
          return int(a/2);


class Problem3(Problem):
    def __init__(self):
        statement = '3. Primiti o stiva. Operatii: \n'
        statement += 'numar -> se inseaza numarul in stiva \n'
        statement += 'P -> se extrage un numar din stiva si se afiseaza \n'
        statement += 'Gasiti o succesiune de mutari a.i. introducand el. 1 2 3 4 5 in stiva (in aceasta ordine) la final sa se afiseze 3 2 4 5 1.'
        statement += '\n\n\n'
```

```
def solve(self):
	solution = 'Vom parcurge vectorul\n'
	solution += 'Se va citi in ordine urmatoarele numere: \n'
	n = len(data);
	print(data);
	stack = [];
	arr = [];
	ok = 0; #ok := verifica daca s-a citit pana la prima jumatate
	i = 0;
	contor = 1;
	n = n-1;
	while i <= n: #O(n^2)
			if i <= div2(n):
				if ok == 0:
					e = data[i]
					numar(e,stack)
					solution +='S-a introdus in stiva numarul ' + str(e) + ' lungimea fiind ' + str(len(stack)) + '\n'
					if i == div2(n):
							ok = 1;
							i = 1;
					else:
						i = i + 1;
						contor += 1;



				elif contor != 1:
					solution += 'Se va elimina din stiva numarul ' + str(data[contor-1]) + '\n'
					p(arr,stack);
					i = i + 1;
					contor -= 1
				if ok == 1 & contor == 1:
					contor = div2(n) + 1;

			else:
				if contor == n:
				    break;
				e = data[contor];
				numar(e,stack);
				solution +='S-a introdus in stiva numarul ' + str(e) + '\n'
				solution += 'Se va elimina din stiva numarul ' + str(data[contor-1]) + '\n'
				p(arr,stack);
				contor = contor + 1;
				i = i + 1;

	solution += 'Se va elimina din stiva numarul ' + str(stack[0]) + '\n'
	p(arr,stack);

	solution += 'Rezultatul final este : ' + str(arr) + '\n'
	return(solution);
```