Programarea Algoritmilor
– SEMINAR NR. 2 –
(grupele 132-134, 141-144)

1. Se citește un număr natural *n*. Să se afișeze toate numerele prime mai mici sau egale decât *n*. (Folosim ciurul lui Eratostene.)

```
# Solutia 1 ---> O(n*n)
n = int(input("n = "))
A = [1] * (n+1)
A[0] = A[1] = 0
for i in range(2, int(n**0.5) + 1):
    if A[i] == 1:
        for j in range(i*i, n+1, i):
            A[j] = 0
print("Numerele prime mai mici sau egale decat", n, "sunt:")
for i in range(2, n+1):
    if A[i] == 1:
        print(i, end=" ")

# Solutia 2 ---> O(n * log(log(n)))
n = int(input("n = "))
A = [2]
for i in range(3, n+1, 2):
    ok = True
    for x in A:
        if i%x == 0:
            ok = False
            break
    if ok == True:
        A.append(i)
print("Numerele prime mai mici sau egale decat", n, "sunt:")
print(*A)
```

2. Se citește un număr natural nenul *n*.
   a) Să se afișeze al *n*-ulea număr din șirul Fibonacci.
   b) Să se afișeze primele *n* numere din șirul Fibonacci.

   a) **TEMĂ:** rezolvare folosind O(1) spațiu (se reține doar două valori din șir) și O(n) timp.

```
n = int(input("n = "))
f1 = f2 = 1
if n == 1:
    print("Primul numar din sirul Fibonacci este: ", f1)
elif n >= 2:
    for i in range(n-2):
        f1, f2 = f2, f1+f2
    print(f"Al {n}-lea numar din sirul Fibonacci este: {f2}")
```

   b) Rezolvare în O(n) spațiu și O(n) timp.

```
n = int(input("n = "))
Fib = [1, 1]
for i in range(n-2):
    f = Fib[-2] + Fib[-1] # suma ultimelor 2 nr din lista
    Fib.append(f)
print("Primele", n, "numere din sirul Fibonacci sunt:")
print(*Fib[:n])
```

3. Se citesc $2k - 1$ numere naturale nenule. Fiecare număr apare de două ori, cu excepția unuia. Să se afișeze numărul care apare o singură dată în șir.

```
k = int(input("k = "))
nr = int(input("nr: "))
for i in range(2*k - 2):
    x = int(input("nr: "))
    nr = nr ^ x
print("\nNumarul care a aparut o singura data:", nr)
```