#!/usr/bin/env python3﻿# -\*- coding: utf-8 -\*-﻿import argparse, sys, os﻿from collections import Counter﻿import string﻿# Алфавити: латиница + кирилица (малки/главни)﻿LAT\_LO = "abcdefghijklmnopqrstuvwxyz"﻿LAT\_UP = LAT\_LO.upper()﻿CYR\_LO = "абвгдежзийклмнопрстуфхцчшщъьюя"﻿CYR\_UP = CYR\_LO.upper()﻿def shift\_char(ch, n):﻿    # латиница﻿    if ch in LAT\_LO:﻿        i = (LAT\_LO.index(ch) + n) % len(LAT\_LO)﻿        return LAT\_LO[i]﻿    if ch in LAT\_UP:﻿        i = (LAT\_UP.index(ch) + n) % len(LAT\_UP)﻿        return LAT\_UP[i]﻿    # кирилица﻿    if ch in CYR\_LO:﻿        i = (CYR\_LO.index(ch) + n) % len(CYR\_LO)﻿        return CYR\_LO[i]﻿    if ch in CYR\_UP:﻿        i = (CYR\_UP.index(ch) + n) % len(CYR\_UP)﻿        return CYR\_UP[i]﻿    # друго не пипаме﻿    return ch﻿def caesar(text, n):﻿    return "".join(shift\_char(ch, n) for ch in text)﻿def read\_text(path):﻿    if not path:﻿        return sys.stdin.read()﻿    with open(path, "r", encoding="utf-8", errors="ignore") as f:﻿        return f.read()﻿def write\_text(path, data):﻿    if not path:﻿        sys.stdout.write(data)﻿        return﻿    with open(path, "w", encoding="utf-8") as f:﻿        f.write(data)﻿# прост честотен анализ (латиница и кирилица) за отгатване на shift при шифър на Цезар﻿# Стъпка: търсим n, при който честотите се доближават до референтни﻿ENG\_FREQ = "etaoinshrdlcumwfgypbvkjxqz"﻿BUL\_FREQ = "оаеиинтрслвкмдпубягхзйчжцшщфьюяъ"  # ориентировъчно﻿def score\_english(s):﻿    s = "".join([c.lower() for c in s if c.isalpha() and c.lower() in LAT\_LO])﻿    counts = Counter(s)﻿    common = "".join([p[0] for p in counts.most\_common(8)])﻿    # пресичане с топ-8 от ENG\_FREQ﻿    return sum(c in ENG\_FREQ[:8] for c in common)﻿def score\_bulgarian(s):﻿    s = "".join([c.lower() for c in s if c.isalpha() and c.lower() in CYR\_LO])﻿    counts = Counter(s)﻿    common = "".join([p[0] for p in counts.most\_common(8)])﻿    return sum(c in BUL\_FREQ[:12] for c in common)﻿def auto\_guess(text):﻿    # Пробваме всички n по отделно за латиница (26) и кирилица (30)﻿    # Връщаме това n (‑25..+25 за латиница; ‑29..+29 за кирилица), което дава най‑висок скор.﻿    best = (None, -1, "latin")  # (n, score, script)﻿    for n in range(26):﻿        cand = caesar(text, -n)  # assume text is encrypted; try to decode﻿        sc = score\_english(cand)﻿        if sc > best[1]:﻿            best = (-n, sc, "latin")﻿    for n in range(30):﻿        cand = caesar(text, -n)﻿        sc = score\_bulgarian(cand)﻿        if sc > best[1]:﻿            best = (-n, sc, "cyr")﻿    return best  # n, score, script﻿def main():﻿    ap = argparse.ArgumentParser(description="Caesar (ROT‑N) tool for Latin & Cyrillic. Works in Termux.")﻿    mode = ap.add\_mutually\_exclusive\_group(required=True)﻿    mode.add\_argument("-e", "--encrypt", action="store\_true", help="Шифриране")﻿    mode.add\_argument("-d", "--decrypt", action="store\_true", help="Дешифриране")﻿    mode.add\_argument("-a", "--auto", action="store\_true", help="Авто‑дешифриране (честотен анализ)")﻿    ap.add\_argument("-n", "--shift", type=int, default=13, help="Отместване N (по подразбиране 13)")﻿    ap.add\_argument("-i", "--input", help="Входен файл (ако липсва – чете от STDIN)")﻿    ap.add\_argument("-o", "--output", help="Изходен файл (ако липсва – пише към STDOUT)")﻿    args = ap.parse\_args()﻿    text = read\_text(args.input)﻿    if args.auto:﻿        n, sc, script = auto\_guess(text)﻿        if n is None:﻿            print("Не успях да отгатна отместването.", file=sys.stderr)﻿            sys.exit(2)﻿        # при auto винаги правим decode с намерения n﻿        result = caesar(text, n)﻿        print(f"[auto] отгатнат shift: {n} (script={script}, score={sc})", file=sys.stderr)﻿        write\_text(args.output, result)﻿        return﻿    # encrypt/decrypt﻿    n = args.shift % 30  # максимална база за кирилица; за латиница модул 26 се хваща автоматично﻿    if args.decrypt:﻿        n = -n﻿    result = caesar(text, n)﻿    write\_text(args.output, result)﻿if \_\_name\_\_ == "\_\_main\_\_":﻿    main()