Zadaci za Laboratorijsku vježbu 2

NAPOMENA: Studenti bi trebali da razmisle o zadacima koji će se raditi na laboratorijskoj vježbi prije nego što dođu na vježbu, tako da već u startu imaju osnovne ideje kako riješiti zadatke. U suprotnom, rad na vježbi neće biti produktivan. Zadatke koje studenti ne stignu uraditi za vrijeme vježbe, trebali bi ih samostalno uraditi kod kuće.

1. Prosti brojevi su oni prirodni brojevi koji imaju *tačno dva različita djelioca* u skupu $\mathbb{N}$: jedinicu i samog sebe (prividno slična definicija po kojoj su prosti brojevi oni i samo oni koji su djeljivi samo sa jedinicom i sa samim sobom nije u potpunosti tačna, razmislite zašto). Napišite prvo funkciju "`DaLiJeProst`" sa jednim parametrom "`n`" koji predstavlja neki cijeli broj, a koja kao rezultat vraća logičku vrijednost "tačno" ili "netačno" u ovisnosti od toga da li je parametar prost broj ili ne. Zatim napišite funkciju "`ProstiBrojeviUOpsegu`", također sa dva cjelobrojna parametra "`a`" i "`b`" koja kao rezultat vraća vektor koji se sastoji od svih prostih brojeva u intervalu od "`a`" do "`b`" uključivo (tj. svih prostih brojeva koji su veći ili jednaki od "`a`", a manji ili jednaki od "`b`"). Napisanu funkciju iskoristite u glavnom programu koji traži da se sa tastature unesu dva cijela broja, a zatim za sve cijele brojeve u opsegu između ta dva unesena broja ispisuje koji su prosti brojevi u datom opsegu, ili informaciju da nema prostih brojeva u datom opsegu. Dijalozi između korisnika i programa trebali bi izgledati poput sljedećih (ovisno ima li ili nema prostih brojeva u zadanom opsegu):

```
Unesite pocetnu i krajnju vrijednost: 15 30
Prosti brojevi u rasponu od 15 do 30 su: 17, 19, 23, 29
```

```
Unesite pocetnu i krajnju vrijednost: 24 28
Nema prostih brojeva u rasponu od 24 do 28!
```

   Pretpostavite da će korisnik unijeti smislene podatke, tj. nije potrebno provjeravati da li su zaista uneseni cijeli brojevi.

2. Za slijed brojeva $a_1, a_2, \ldots, a_n$ kažemo da ima period p ukoliko je $1 \leq p < n$ i ukoliko vrijedi da je $a_i = a_{i+p}$ za sve vrijednosti i za koje ova jednakost ima smisla (tj. za koju su oba indeksa i i $i+p$ legalni). Na primjer, slijed brojeva 1, 3, 1, 4, 2, 1, 3, 1, 4, 2, 1, 3 ima period 5, jer je $a_i = a_{i+5}$ za sve vrijednosti i takve da su oba indeksa i i $i+5$ unutar dozvoljenog opsega (tj. za i od 1 do 7 uključivo). Isti slijed također ima i period 10. Dalje, za slijed brojeva kažemo da je periodičan ukoliko postoji makar jedan broj p koji je period tog slijeda, pri čemu najmanji takav broj p nazivamo *osnovni period* slijeda. Ukoliko takav broj p ne postoji, slijed nije periodičan. Na primjer, gore navedeni slijed brojeva je periodičan s osnovnim periodom 5, dok slijed brojeva 4, 5, 1, 7, 1, 5 nije periodičan. Vaš zadatak je da napišete dvije funkcije "`TestPerioda`" i "`OdrediOsnovniPeriod`". Prva funkcija prima dva parametra, pri čemu je prvi parametar vektor realnih brojeva, dok je drugi parametar cijeli broj p. Funkcija treba da vrati kao rezultat logičku vrijednost "tačno" ili "netačno" ovisno od toga da li slijed brojeva pohranjen u vektoru periodičan sa periodom p ili ne. Druga funkcija prima samo jedan parametar koji je vektor realnih brojeva, a ona kao rezultat vraća osnovni period slijeda pohranjenog u vektoru ukoliko je slijed periodičan, ili nulu ukoliko slijed nije periodičan. To ćete najlakše izvesti tako što ćete iz druge funkcije pozivati prvu funkciju kao pomoćnu, a kako to tačno izvesti, ostavlja Vam se za razmišljanje. Napisane funkcije iskoristite u glavnom programu u kojem ćete unositi elemente sa tastature u neki vektor sve dok se sa tastature ne unese nula, koja označava kraj unosa (tu nulu ne treba smjestiti u vektor). Nakon završetka unosa, program poziva funkciju "`OdrediOsnovniPeriod`" sa ciljem da utvrdi da li se elementi periodično ponavljaju ili ne, nakon čega ispisuje odgovarajući komentar na ekranu (informaciju o dužini perioda, ili da elementi ne čine periodičan slijed). Napomenimo da nije unaprijed poznato koliko će korisnik unijeti elemenata prije nego što unese nulu kao oznaku završetka unosa. Dijalozi između korisnika i programa trebali bi izgledati poput sljedećih (ovisno da li je slijed periodičan ili ne):

```
Unesite slijed brojeva (0 za kraj): 1 3 1 4 2 1 3 1 4 2 1 3 0
Slijed je periodican sa osnovnim periodom 5.
```

```
Unesite slijed brojeva (0 za kraj): 4 5 1 7 1 5 0
Slijed nije periodican!
```

3. Kompleksni brojevi svoju najveću primjenu nalaze upravo u elektrotehnici (i u kvantnoj fizici). Recimo, u teoriji električnih kola uvodi se pojam *impedanse*, koja je kompleksan broj $\bar{Z}$ definiran kao $\bar{Z} = R + Xi$, gdje je R tzv. aktivni otpor, a X tzv. reaktivni otpor (ili reaktansa), koji može biti i negativan. Ukoliko imamo paralelni spoj n elemenata čije su impedanse $\bar{Z}_1, \bar{Z}_2, \ldots, \bar{Z}_n$, ukupna impedansa paralelne veze $\bar{Z}$ računa se kao $\bar{Z} = 1/(1/\bar{Z}_1 + 1/\bar{Z}_2 + \ldots + 1/\bar{Z}_n)$. Drugim riječima, situacija je slična paralelnom spajanju otpornika, osim što se ovdje radi sa kompleksnim impedansama. Napišite program u kojem se korisnika traži da se unese broj elemenata n, a zatim n impedansi $\bar{Z}_k$ za sve k od 1 do n. Impedanse se unose kao kompleksni brojevi, tačnije kao parovi realnih i imaginarnih dijelova unutar zagrada, razdvojeni zarezom. Nakon toga, treba da izračuna i ispiše impedansu paralelne veze svih n elemenata, ponovo kao kompleksan broj. Dijalog između programa i korisnika treba da izgleda poput sljedećeg:

```
Unesite broj elemenata: 3

Z_1 = (3.5,2.8)
Z_2 = (10,-1.54)
Z_3 = (12.37,0.24)

Paralelna veza ovih elemenata ima impedansu Z_ = (2.51479,0.897637).
```

   Za realizaciju programa koristite kompleksni tip podataka. Nemojte koristiti niti nizove niti vektore, nego sumu $1/\bar{Z}_1 + 1/\bar{Z}_2 + \ldots + 1/\bar{Z}_n$ računajte "u hodu", uporedo sa unosom podataka. Također, možete pretpostaviti da će svi uneseni podaci biti smisleni.

4. Prepravite prethodni program tako što će se umjesto impedansi $\bar{Z}_k$ sa tastature posebno unositi aktivni i reaktivni otpori R_k i X_k, i što će se na kraju ispisivati posebno aktivni i reaktivni otpor paralelne veze elemenata. Dijalog između programa i korisnika treba da izgleda poput sljedećeg:

```
Unesite broj elemenata: 3

R1 = 3.5
X1 = 2.8

R2 = 10
X2 = -1.54

R3 = 12.37
X3 = 0.24

Paralelna veza ovih elemenata ima R = 2.51479 i X = 0.897637.
```

5. Prepravite prethodni program tako što će se sve impedanse $\bar{Z}_k$ umjesto preko aktivnog otpora R_k i i reaktivnog otpora X_k zadavati preko tzv. prividnog otpora Z_k i faznog pomaka φ_k, pri čemu vrijedi $\bar{Z}_k = Z_k\, e^{i\varphi_k}$. Program na kraju treba prikazati prividni otpor i fazni pomak za čitavu paralelnu vezu. Fazni pomak treba zadavati i ispisivati u stepenima. Dijalog između programa i korisnika treba da izgleda poput sljedećeg:

```
Unesite broj elemenata: 2

Z1 = 10.5
fi1 = 30

Z2 = 2.8
fi2 = -47.6

Paralelna veza ovih elemenata ima Z = 2.57147 i fi = -33.7613.
```

   Uputa: trebaće vam funkcije "`abs`", "`arg`" i "`polar`".

Za potrebe realizacije zadataka 4. i 5. **dozvoljeno je** kopirati programski kôd zadatka 3.