Izpit iz predmeta Programiranje 1, 23. januar 2018 ob 10.00. Čas pisanja: 90 minut.

Rešitev oddajte prek Učilnice. Za rešitev naloge lahko dobite določeno število točk, **tudi če ne prestane testov**. Funkcija, ki prestane vse teste, **še ni nujno pravilna**. Upošteva se tudi kvaliteta rešitve.

Dovoljena je uporaba vseh materialov na Učilnici in druge literature na poljubnih medijih. Prepovedan je dostop do vseh drugih spletnih strani in vsaka oblika komunikacije, razen s profesorjem oz. asistentom.

1. Migracije

Napiši funkcijo `migracije(ime_datoteke)`, ki prejme ime datoteke, ki vsebuje vrstice oblike

```
8: Maribor -> Ljubljana
3: Maribor -> Nova Gorica
10: Ljubljana -> Maribor
5: Koper -> Nova Gorica
3: Novo mesto -> Nova Gorica
```

Vsaka vrstica pove, koliko ljudi se je preselilo odkod kam. Funkcija naj vrne par imen krajev: kraj, iz katerega je odšlo največ ljudi in kraj, v katerega se je preselilo največ ljudi. V gornjem primeru funkcija vrne `("Maribor", "Nova Gorica")`.

2. Zakladi

Imamo razred `Robot`. Ta se v začetku nahaja na koordinatah (0, 0) in je obrnjen na desno. Koordinatni sistem je takšen kot pri matematiki: koordinata y narašča navzgor. Robot ima naslednje metode.

- `naprej(d)` gre za d naprej v podani smeri;
- `desno()` se obrne za 90 stopinj v desno;
- `levo()` se obrne za 90 stopinj levo;
- `koordinate()` vrne trenutne koordinate (x in y)

Večini je razred znan iz domače naloge; tokrat je že podan v datoteki s testi.

Napisati pa morate funkcijo `zakladi(navodila)`, ki prejme navodila za robota v obliki niza, na primer `"7D3221LL"`, kar bi pomenilo, da skoči za 7 polj naprej, se obrne desno, nato skoči za 3 polja naprej, za 2 polji naprej, za 2 polji naprej, za 1 polje naprej, nato se obrne levo in še enkrat levo. Na vseh poljih, za katera velja, da je **vsota absolutnih vrednosti** njunih koordinat deljiva s 7, je zaklad. (To velja tudi za začetno polje, (0, 0)!).

Funkcija naj vrne, koliko polj z zakladom je robot obiskal. Če večkrat obišče polje z zakladom, dobi zaklad le enkrat. Če zaklad preskoči (torej: gre čez tisto polje, vendar se ne ustavi na njem, ga ne dobi.)

3. Roboti

Recimo, da imamo tri robote in jim damo navodila `"JSVZZVJ"`. Vsi roboti začnejo na (0, 0) in izmenično jemljejo navodila iz niza. Prvi robot bo šel za eno polje na **J**, drugi na **S**, tretji na **V**, potem spet prvi na **Z**, drugi **Z**, tretji **V**, prvi **J**. Njihove koordinate po tem so `[(-1, -2), (-1, 1), (2, 0)]`. Prvi, recimo, je šel namreč na J, Z in J, zato (-1 in -2).

Napiši funkcijo `roboti(navodila, n)`, ki prejme navodila in število robotov (niso nujno trije!), ter vrne seznam terk njihovih koordinat.

4. Brez ponavljanja

Napiši **rekurzivno** funkcijo `brez_ponavljanja(s)`, ki prejme seznam in vrne nov seznam, v katerem vse zaporedne ponovitve istega elementa zamenja z enim samim elementom. Klic `brez_ponavljanja([3, 1, 1, 4, 2])` vrne `[3, 1, 4, 2]`.

Pomoč: Seznam brez ponavljanja dobimo tako, da prvi element obdržimo, ali pa ne. Slediti pa mora seznam brez ponavljanja.

OBRNI LIST

5. Delitelj

Napiši razred `Delitelj`, katerega **konstruktor** kot argument prejme število; imenujmo ga `k`.

Razred ima **metodo** akcija(s), ki prejme seznam. V njem vsa števila, ki so deljiva s `k`, deli s `k`.

Tega ne dela zastonj: za vsako deljenje zaračuna toliko, kolikor je velik količnik. Skupno ceno vseh dosedanjih klicev akcij hrani v **atributu** `cena`.

```
>>> s = [25, 8, 9, 15, 3, 81]
>>> deli5 = Delitelj(5)
>>> deli5.akcija(s)
>>> s
[5, 8, 9, 3, 3, 81]
>>> deli5.cena
8
>>> deli5.akcija(s)
>>> s
[1, 8, 9, 3, 3, 81]
>>> deli5.cena
9
```

Prva cena je 8, ker je naračunal 25/5 = 5 in 15/5 = 3, in 5 + 3 je enako 8. V drugem klicu se cena poveča za 1, ker je delil 5 z 1 in dobil 1.