Zadatak

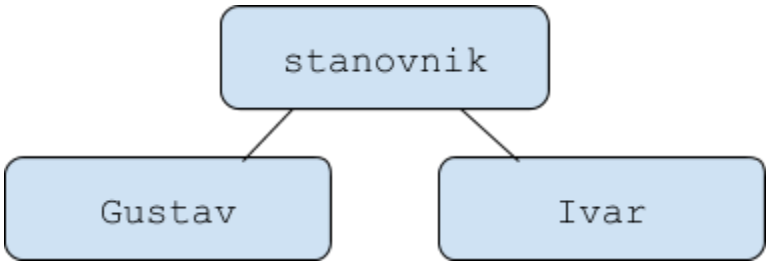
U malom mjestu u Skandinaviji na snazi su mjere opreza zbog epidemije. Stanovnici su karakterno ili Gustavi (oprezno se drže distance) ili Ivari (važnija im je socijalna komponenta od zdravstvenog rizika).

Naselje je predstavljeno 100 x 100 pločom, koordinate su prirodni brojevi od 0 do 99. Svi stanovnici se kreću kao kralj u šahu, unutar naselja (ne mogu ići preko ruba). Na svakom polju može biti više od 1 stanovnika. Udaljenost stanovnika na poljima (a,b) i (c,d) je najmanji broj koraka/poteza potreban da se dođe od jednog do drugog. Npr. udaljenost polja (1,1) i (5,8) je 7.

Svaki stanovnik vidi gdje su ostali stanovnici i može za svake koordinate (a,b) izračunati funkciju **razmaknutost**(a,b)=zbroj_udaljenosti_ od_(a,b)_do_svih_ostalih_stanovnika - 50*broj_stanovnika_na susjednim_poljima_od(a,b) - 1000*broj_stanovnika_na(a,b). (Sebe ne računa u toj formuli). Gustavi žele biti na polju gdje je razmaknutost što veća, a Ivari na polju gdje je razmaknutost što manja.

Napravite sljedeću hijerarhiju klasa koja opisuje njihovo ponašanje.

Sučelje spremite u datoteku **stanovnik.h**, a implementaciju spremite u datoteku **stanovnik.cpp.** Sve funkcije i strukture smiju biti javne.



`stanovnik` neka je apstraktna klasa. Implementirajte sljedeće funkcije članice:

- konstruktor(i) – prima(ju) koordinate stanovnika (int, int) i matični broj (pozitivan `int`), ili samo koordinate, i tada se stanovniku dodijeljuje prvi slobodan matični broj, počev od 1. Neće biti primjera sa ponavljanjima matičnih brojeva.
- `xK(), yK(), mat_broj()` - nemaju parametara, vraćaju odgovarajuće informacije o stanovniku
- `stanovnik& najblizi()` - članica stanovnika, vraća referencu na najbližeg stanovnika, ako ih ima više, onoga sa najmanjim matičnim brojem.
- `stanovnik& fokusiraj( stanovnik& B)` - stanovnik koji je pozvao funkciju mijenja svoj položaj u odnosu na položaj stanovnika B, u skladu sa svojim karakterom, za korak bliže ili korak dalje od B. Gustav se, neovisno o *razmaknutosti*, pokušava udaljiti od B (povećati broj koraka do B), dok se Ivar pokušava približiti B. (Kažemo "pokušava" jer udaljavanje može biti nemoguće ako je objekt koji je pozvao već u kutu a približavanje nemoguće ako su već na istom polju.) Ako imamo više mogućih koraka, gledamo i euklidsku-2 udaljenost.
  Neće biti poziva sa dvosmislenim kandidatima za odgovor (npr Gustav se odmiče od Ivara a imaju bar jednu koordinatu istu).
  Funkcija neka vraća referencu na objekt koji ju je pozvao.
- `int razmaknutost(int a,int b)` – stanovnik računa razmaknutost polja (a,b) u odnosu na položaj ostalih stanovnika. Ne računa stanovnika koji ju je pozvao.

- `bool lokOptimiziraj()` – stanovnik koji je pozvao, ili ostane na istom polju ili se pomakne na neko od susjednih, ako je tamo razmaknutost poželjnija (strogo bolja). Ako ima više lokalno boljih polja, pomiče se na ono sa manjom x koordinatom, a ako su i x kordinate jednake, na ono sa manjom y kordinatom. Ako se nije pomakao, vraća false, u protivnom vraća true.
- `void lokOptSve()` – `static` funkcija, svakom stanovniku lokalno optimizira poziciju ~~svih stanovnika~~, po redu nastanka (od starijeg prema novijem). Proces nije simultan, prvo se pomjeri najstariji stanovnik, onda se drugi najstariji pomiče s obzirom na novonastalu situaciju itd.
- `Svi () - static` funkcija koja vraća listu pokazivača na sve stanovnike koji trenutno postoje, sortiranu prema vremenu nastanka od starijeg prema novijem.

Možete staviti `public` pristup u svim klasama za sve varijable i funkcije. Više stanovnika se može nalaziti na istom polju. Ako za funkciju nije napomenuto da je specifična za izvedenu klasu, znači da se odnosi na baznu klasu. Gore navedeni su **nužni** elementi strukture, smijete imati i druge dodatne. najblizi, lokOptimiziraj, lokOptSve neće biti pozivane u situaciji gdje postoji samo jedan stanovnik.

Opće napomene

- Struktura, funkcije i datoteke moraju se zvati točno onako kako je zadano u zadatku. Pazite na mala i velika slova.
- Trebate poslati samo sučelje i implementaciju. U datoteci koju šaljete ne smije se nalaziti funkcija main()!
- Nijedna funkcija ne smije ništa učitavati s tipkovnice ili neke datoteke, niti išta ispisivati na ekran ili neku datoteku.
- Svaki od mainova kojim testiramo vaše programe neće pozivati sve navedene funkcije. Stoga, ako neku funkciju ne znate napisati, ipak možete skupiti koji bod na funkcije koje znate implementirati (u tom slučaju funkciju koju ne znate nemojte navesti u .h datoteci ili napravite neku trivijalnu implementaciju)

Ispravnost implementacija koje napišete bit će provjerena tako da ćemo mi napisati razne klijentske programe koji će deklarirati nekoliko varijabli zadane strukture, i na njima pozivati funkcije koje ste trebali napisati. Ako se poslani programi ne budu uspješno povezivali (linkali) s našim klijentskim programima, smatrat će se neispravnima.

Neki klijentski programi provjeravat će samo neke jednostavnije funkcije, dok će neki provjeravati sve funkcije koje trebate napisati. Provjera je potpuno automatska, tako da je od presudne važnosti da se pridržavate specifikacije. Nepridržavanje lako može uzrokovati osvojenih 0 bodova iz zadaće!

Upiti u vezi zadaće na mail *marijan.polic@math.hr*

Primjer maina:

```
#include<iostream>
#include<list>
#include”stanovnik.h”

using namespace std;
int main(){
        Ivar I1(3,5,5), I2(4,7,6), I3(20,20);
```

```
        Gustav G1(4,9), G2(11, 0), G3(30,30),G(12,12 );
        list<stanovnik*>::iterator i;
        list<stanovnik*> S;

        cout<<G3.mat_broj()<<" na koord:"<<G3.xK()<<","<<G3.yK()<<endl;   //4 na koord:30,30


cout<<G1.razmaknutost(4,9)<<","<<G3.razmaknutost(25,25)<<","<<G1.razmaknutost(3,5)<<endl;
//65,107,-937

        cout<<G.najblizi().mat_broj() <<" na
koord:"<<G.najblizi().xK()<<","<<G.najblizi().yK()<<endl;   //1 na koord:20,20

        G3.lokOptimiziraj(); I1.lokOptimiziraj();
        cout<<G3.mat_broj()<<" na koord:"<<G3.xK()<<","<<G3.yK()<<endl;  //4 na koord:31,31
        cout<<I1.mat_broj()<<" na koord:"<<I1.xK()<<","<<I1.yK()<<endl;  //5 na koord:4,6
//I1 dospije na polje koje je susjedno sa I2, koje ima za njega puno povoljniju (manju)
razmaknutost


        G1.fokusiraj(G3).fokusiraj(G3);
        cout<<G1.mat_broj()<<" na koord:"<<G1.xK()<<","<<G1.yK()<<endl;  //2 na koord:2,7

  S=G1.Svi();

        stanovnik::lokOptSve(/*bez arg. 1*/);

        for(i=S.begin();i!=S.end();++i) cout<<(*i)->mat_broj()<<" na
koord:"<<(*i)->xK()<<","<<(*i)->yK()<<endl; /*
5 na koord:4,7
6 na koord:4,7
1 na koord:19,19
2 na koord:1,6
3 na koord:12,0
4 na koord:32,32
7 na koord:13,11
*/
}
```