Zadaća 4
Objektno orijentirano programiranje

Decembar, 2018

Sadržaj

1 Problem 1

Minesweeper

Cilj igre je naći gdje se nalaze sve mine unutar `M x N` polja. Da bi vam pomogla igra vam pokaže broj unutar ćelije koji govori koliko se mina nalazi u neposrednoj blizini te iste ćelije. Na primjer, pretpostavimo da imamo polje `4 x 4` sa 2 mine (predstavljene karakterom '*'):

```
*...
....
.*..
....
```

Ukoliko bi predstavili isto polje prema gore navedenom primjeru izgledalo bi:

```
*100
2210
1*10
1110
```

Kao što možete primjetiti, svaka ćelija može imati najviše 8 susjednih polja.

Ulaz
Ulaz se sastoji od proizvoljnog broja polja. Prva linija svakog polja sastoji se od dva cijela broja m i n koji pretstavljaju broj linija i kolona polja respektivno. Svaka sigurna ćelija predstavljena je karakterom '.' (bez navodnika), dok su mine predstavljene karakterom '*' (bez navodnika). Prva linija polja u kojoj je `m = n = 0` predstavlja kraj unosa i ne bi trebala biti procesirana.

Izlaz
Za svako polje potrebno je u zasebnoj liniji ispisati:

```
Polje #x:
```

Gdje x predstavlja broj polja (počevši od 1). Sljedećih n linija predstavljaju polje u kome je karakter '.' zamijenjen brojem susjednih mina te ćelije. Između svakog polja potrebno je umetnuti praznu liniju.

Primjer ulaza

```
4 4
*...
....
.*..
....
3 5
**...
.....
.*...
0 0
```

Primjer izlaza

```
Polje #1:
*100
2210
1*10
1110

Polje #2:
**100
33200
1*100
```

Napomena

- Za implementaciju elementa polja (ćelije), implemenirati klasu `Celija` koja sadrži infomacije o svojoj poziciji unutar polja te broj mina u susjednim poljima, ukoliko polje nije mina.
- Za implementaciju polja implementirati klasu `Polje` koja dinamički alocira dvodimenzionalni niz Celija, u odnosu na input korisnika.
- Dodatno, implementirati klasu `Minesweaper` koja sadrži kontejner Polja, te implementira metod.

  ```
  void Minesweaper::pokreni() {
      parsiraj();
      ispisi();
  }
  ```

  Prilagoditi navedene metode, tako da omoguće izvršavanje pograma, kao što je opisano u zadatku. `main` u osnovi treba da izgleda kao:

```
Minesweaper igra;
try {
    igra.pokreni();
} catch {

}
```

- Voditi računa o memory leaku!

2 Problem 2

Implementirati klasu `Lista` koja implementira dvostruko povezanu listu. Lista je skup čvorova koji se sastoje od podataka istog tipa. Potrebno je implementirati listu cijelih brojeva. Za definiciju čvora koristiti strukturu, gdje svaki čvor ima pokazivač na sljedeći i prethodni element liste. Za klasu `Lista` potrebno je definisati:

- `Lista() = default; //default constructor`
- `Lista(const Lista&); // copy constructor`
- `Lista& operator=(const Lista&); // assignment operator`
- `~Lista(); // destructor`
- `Lista(std::initializer_list<int> a);`

Dodatno, implementirati:

- Dodavanje elemenata u listu, na poziciju definisanu indeksom, prije elementa koji se nalazi na specificiranoj poziciji.
  `void Lista::insert(size_t, int);`
  Ukoliko je indeks van granica, baciti odgovarajuću grešku.
- Dodavanje elemenata na kraj liste
  `void Lista::push_back(int);`
- Dodavanje elemenata na početak liste
  `void Lista::push_front(int);`
- Brisanje elementa iz liste, sa pozicije definisane indeksom
  `void Lista::erase(size_t);`
  Ukoliko je indeks van granica, baciti odgovarajuću grešku.

- Metod `size_t size();` koji nalazi dužinu liste, odnosno broj elemenata.