OBJEKTNO ORIJENTISANO PROGRAMIRANJE

- domaći zadatak broj 1 -

Funkcionalna specifikacija

Na programskom jeziku C++ implementirati skup klasa za igru „Jamb" za jednog igrača.

Specifikacija klase Game

Klasa `Game` predstavlja klasu za pokretanje, čuvanje i učitavanje igre.

Propisno stvaranje i uništavanje

Implementirati propisno stvaranje i uništavanje objekata klase `Game`.

Pravila igre „Jamb"

Jamb igra se igra sa 5 ***kockica***. Cilj igre je dobiti što više poena bacanjem kockica, pri čemu različite kombinacije nose različit broj poena. Igrač baca kockice najviše 3 puta. Posle svakog ***bacanja*** određuje se koje se kockice čuvaju (zadržavaju), a koje ponovo bacaju. Nakon poslednjeg bacanja, igrač upisuje rezultat u ***tabelu*** sa poenima. U jednom potezu igrač mora da upiše rezultat u tačno jedno polje tabele (čak iako nema šta da upiše, u tom slučaju bira polje u koje će da upiše 0 poena), nakon čega počinje nov potez.

	↓	S	↑
1			
2			
3			
4			
5			
6			
∑			
max			
min			
∑			
KENTA 66, 56, 46			
TRILING +20			
FUL +30			
POKER +40			
YAMB +50			
∑			

Tabela sadrži tri kolone („Nadole", „Slobodna", „Nagore") i određen broj redova. Kolona „Nadole" mora da se popunjava redom od prvog do poslednjeg reda. Kolona „Nagore" mora da se popunjava redom od poslednjeg do prvog reda. Kolona „Slobodna" može da se popunjava proizvoljnim redosledom.

Prvih 6 redova tabele odvojeno je za upis poena koji predstavljaju zbir kockica na kojima se nalazi odgovarajući broj (npr. u red „1" se upisuje zbir bačenih kečeva, u red „2" se upisuje zbir bačenih dvojki, itd.).

Nakon prvih 6 redova, slede redovi „Max" i „Min". U redove „Max" i „Min" se upisuje zbir svih 5 kockica. Redovi „Max" i „Min" se na kraju partije oduzimaju za svaku kolonu zasebno i množe sa brojem upisanih kečeva u toj koloni (npr. ako je u polje „Max" upisan broj 28, u polje „Min" upisan broj 7, a u polje „1" upisan broj 4, ukupan broj poena u ovoj sekciji je (28-7)*4=84).

Poslednjih nekoliko redova nastalo je po ugledu na igru „Poker":

U polje „Kenta" upisujemo broj 66, 56 ili 46, u zavisnosti da li se „Kenta" dobila u prvom, drugom ili trećem bacanju. U polje „Kenta" smemo da upišemo vrednost samo ako je igrač na kockicama dobio pet uzastopnih vrednosti.

U polje „Triling" smemo da upišemo vrednost ako je igrač na kockicama dobio tri ista broja. Vrednost koja se upisuje je zbir vrednosti ta tri broja uvećan za 20.

U polje „Ful" smemo da upišemo vrednost ako je igrač na kockicama dobio kombinaciju od tri ista i dva ista broja. Vrednost koja se upisuje je zbir vrednosti tih pet brojeva uvećan za 30.

U polje „Poker" smemo da upišemo vrednost ako je igrač na kockicama dobio četiri ista broja. Vrednost koja se upisuje je zbir vrednosti ta četiri broja uvećan za 40.

U polje „Jamb" smemo da upišemo vrednost ako je igrač na kockicama dobio pet istih brojeva. Vrednost koja se upisuje je zbir vrednosti tih pet brojeva uvećan za 50.

Format tekstualne reprezentacije tabele igre

Tekstualna reprezentacija tabele igre se sastoji iz 13 redova u kojima se nalaze upisane vrednosti u odgovarajuće ćelije tabele, pojedinačno odvojene zarezima unutar jednog reda, pri čemu vrednosti nisu obavezne. U svakom redu nalaze se maksimalno tri vrednosti. Primer prva četiri reda tekstualne reprezentacije tabele igre se nalazi u nastavku:

```
5,3,
8,,
9,,3
20,16,8
```

Učitavanje i čuvanje igre

Implementirati operaciju `void Game::load(string table);` koja učitava igru čija je tabela sa rezultatima data parametrom operacije. Ukoliko je prethodno već učitana igra, sve informacije o staroj igri se brišu i učitava se nova igra. Ukoliko format tabele nije dobar, ili je u neko polje upisana vrednost koja nije odgovarajuća za dato polje, ignorisati učitavanje igre i ispisati grešku na standardnom izlazu.

Implementirati operaciju `string Game::save();` koja kao povratnu vrednost treba da vrati tabelu sa rezultatima.

Implementirati operaciju `void Game::restart();` koja učitava novu igru sa praznom tabelom sa rezultatima.

Tok igre

Implementirati operaciju koja vrši bacanje kockica. Nakon trećeg poziva funkcije onemogućiti ponovni poziv sve dok se u tabelu sa rezultatima ne upiše nov rezultat. Pozivom ove funkcije bacaju se samo kockice koje korisnik nije sačuvao (zadržao). U prvom bacanju bacaju se uvek sve kockice. Na kraju poziva ove funkcije, vrednosti svih kockica (i zadržanih i bačenih) ispisuju se na standardni izlaz.

```
void Game::throwDice();
```

Implementirati operaciju koja postavlja kockice na vrednosti prosleđene kroz niz od tačno pet elemenata. Ova operacija nije deo igre, već se koristi za testiranje.

```
void Game::setDice(int dice[]);
```

Implementirati operaciju koja čuva (zadržava) odabrane kockice. Parametar funkcije je niz od tačno pet `boolean` vrednosti koje odgovaraju kockicama i određuju da li određenu kockicu korisnik želi da zadrži ili ne. U slučaju prosleđivanja nekorektnog niza, povratna vrednost funkcije je `false`. Redosled kockica unutar niza odgovara redosledu ispisa kockica nakon njihovog bacanja.

```
bool Game::keepDice(bool dice[]);
```

Implementirati operaciju koja ispisuje sve opcije koje je moguće upisati u tabelu sa rezultatima (uključujući i polja u koje je moguće upisati nulu). Ova operacija ne može da se pozove pre prvog bacanja kockica. Redosled opcija odgovara čitanju tabele po kolonama.

```
void Game::printChoices();
```

Implementirati operaciju koja upisuje rezultat u tabelu. Parametrom se zadaje pozicija ćelije u koju korisnik želi da upiše rezultat, pri čemu je redosled opcija određen ispisom opcija iz prethodne operacije i kreće od 0. U slučaju prosleđivanja nekorektne pozicije, povratna vrednost funkcije je `false`.

```
bool Game::writeResult(int choice);
```

Implementirati operaciju koja ispisuje trenutnu tabelu sa rezultatima.

```
void Game::printResults();
```

Implementirati operaciju koja vraća trenutni ukupan broj poena.

```
int Game::getScore();
```

Implementirati operaciju koja ispituje da li je igra završena.

```
bool Game::isGameOver();
```

Test funkcija

Javni test je predstavljen fajlom „tst.h" koji testira igru. Studentima je javno dostupna implementacija svih funkcija u datom fajlu i mogu da menjaju postojeće funkcije ili da dodaju nove funkcije da bi dodatno testirali svoj kod na način koji će biti objašnjen na auditornim vežbama. Glavna funkcija se nalazi u fajlu „source.cpp" i redom poziva sve funkcije iz fajla sa testovima.

Tehnički zahtevi i smernice za izradu rešenja

Iz kolekcije standardne biblioteke dozvoljeno je koristiti SAMO tip podataka *string*. NIJE DOZVOLJENO koristiti kolekcije iz standardne biblioteke, već isključivo ugrađene i sopstevene tipove podataka i strukture. Sve klase i operacije moraju biti imenovane prema zahtevima iz domaćeg zadatka. Ukoliko u zadatku nešto nije dovoljno jasno definisano, treba usvojiti razumnu pretpostavku i na temeljima te pretpostavke nastaviti izgrađivanje svog rešenja. Za uspešno odbranjen domaći zadatak potrebno je na odbrani pokazati kod podeljen na odgovarajuće .h i .cpp fajlove. Rok za predaju rešenja biće naknadno saopšten.

06. 11. 2023. godine sa predmeta