Univerzitet u Novom Sadu

Fakultet tehničkih nauka

Moj broj

Analiza problema igre

Student: Luka Popović

Predmet: Objektno orijentisano programiranje 2

Tema projektnog zadatka: Igra moj broj

# **Opis problema**

Rad U/I podsistema

Podaci se učitavaju iz fajla „UlaznaDadoteka.txt“. Svaka linija tog fajla sadrzi 4 jednocifrena broja, 2 dvocifrena i jedan trocifren broj. Trocifren broj predstavlja traženi broj, a ostalih 6 brojevi pomoću kojih treba doći do traženog broja.

Svi brojevi su odmaknuti space, dok se linija završava sa tab.

Brojevi su smesteni u vektor vektora, tako sto jedan vektor predstavlja liniju jednu, a vektor unutar tog vektora predstavlja brojeve.

Pisanje podataka se vrši u fajlu „Rezultati.txt“. U fajl se upisuje nakon svake runde, tako što se upisuje broj runde, pobednik i druge stvari date u specifikaciji.

Na kraju dadoteke se upisuje ko je pobedio kojim rezultatom.

Spisak svih klasa, izuzetaka i slobodnih funkcija

Klasa je TokenStream i struktura Token.

Kalkulator.h i Kalkulator.cpp su funkcije:

double get\_value(string);

void set\_value(string, double);

bool is\_declared(string );

double primary();

double term();

double expression();

double declaration();

double statement();

void clean\_up\_mess();

double calculate();

vector<double> korisceniBrojevi();

void brisanjeBrojevi();

RacunanjeIzraza.h i RacunanjeIzraza.tpp funkcije:

string concat(string&, string&, char);

map<long long, string> all\_values(int, int);

void solve();

void ucitavanje\_file(ifstream& );

void read\_ucitaj();

string izraz\_resenje();

Objašnjenje najbitnijih atributa, klasa i funkcija članica, slobodnih funkcija i izuzetaka

Klasa je TokenStream i struktura Token su rađene na vežbama 3 (Advanced Calculator) tako da ne bi puno ulazio u analizu istih.

Ono što je promenjeno u odnosu na to je što je razdvojen kod u Token.h i Token.cpp i isto tako u TokenStream.h i TokenStream.cpp, a oni se grupišu u Calculator.h i Calculator.cpp.

U RačunanjeIzraza.h imaju sledeće funkcije:

string concat(string&, string&, char);

map<T, string> all\_values(T, T);

void ucitavanje\_file(ifstream&);

void read\_ucitaj();

void solve();

string izraz\_resenje();

Prva funkcija vraća string tako što stavlja u zagrade izraz i operator između njih.

Druga funkcija služi za pravljenje svih mogućih izraza a njena povratna vrednost je mapa čiji je ključ vrednost izraza, a vrednost izraz u obliku stringa.

Treća funkcija učitava fajl i stavlja vrednosti dok četvrta funkcija razdvaja traženi broj I brojeve s kojim igrač može da igra.

Peta funkcija je glavna za algoritam, u njoj se prave sve moguće permutacije brojeva i u njoj se poziva druga funkcija, koja pravi tačan izraz. Ako se ne može naći tačno rešenje, naćiće prvo najbliže.

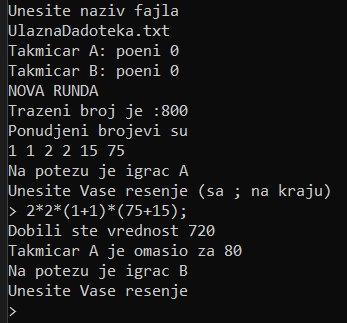
Šesta funkcija samo vraća izraz kao string jer je potrebna prilikom upisa u Rezultati.txt.

Strukture argumenata komandne linije i primer korišćenja

Na početku se unosi naziv fajla koji treba da je ulazni. Ako taj fajl ne postoji izaćiće iz programa.

Nakon toga takmičar A i takmičar B mogu unositi njigova rešenja.

Primer korišćenja:

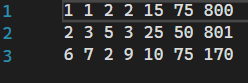


UlaznaDadoteka.txt predstavlja tekstualnu dadoteku u kojoj se nalaze podaci.

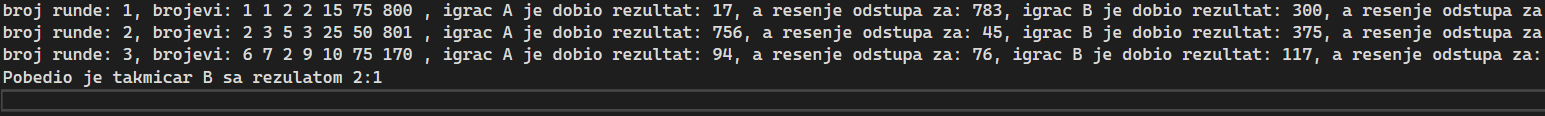
Karakter > predstavlja znak da takmičar treba da unese izraz.

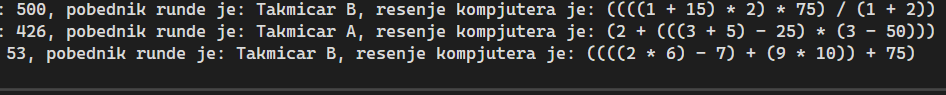
Strukture ulazne i izlazne dadoteke

Ulazna dadoteka: Kao sto je napisano u Rad U/I sistema, u liniji se nalazi 7 brojeva od toga prva 4 predstavljaju jednocifrene brojeve, sledeca dva dvocifrene brojeve i konačno 7 predstavlja traženi broj.



U izlaznu dadoteku se upisuju podaci dati u specifikaciji.





Opis algoritma za pronalaženje tačnog rešenja

Sve funkcije vezane za pronalaženje tačnog rešenja se nalaze u RacunanjeIzraza.h i RacunanjeIzraza.tpp.

Ideja algoritma je u tome da se prave sve moguće permutacije brojeva i da se naprave svi mogući izrazi koja se čuva u mapu čija je ključ vrednost izraza (traženi broj) a vrednost string tj. izraz.

U slučaju da ne može da se nađe tačno rešenje, uvode se dve promenjive (broj koji je veći od traženog broja i broj koji je manji od traženog broja) i u petlji se oni smanjuju, odnosno povećava za jedan dokle god ne naiđe na prvo bliže rešenje i onda se izađe iz programa.

Opis načina testiranja

Za testiranje je napravljen poseban solution Testovi.sln i tu je testirano.

Testirane su 2 metode, računanje izraza i pronalaženje tačnog rešenja.

Sve je testirano u TestoviMain.cpp.

Testiranje za pronalaženje tačnog rešenja se učitava iz tekstualnog fajla i onda se ispisuje tačno rešenje dok se za testiranje za računanje izraza unosi u konzoli.

Uočeni problemi i organičenja

Problem je kada takmičar unosi svoje rešenje mora da unese na kraju ; i ako zaboravi pukne program.

Programu treba više vremena kada ne nadje pravo rešenje.