OBJEKTNO ORIJENTISANO PROGRAMIRANJE

- domaći zadatak broj 1 -

Funkcionalna specifikacija

Na programskom jeziku C++ implementirati statičku biblioteku (.lib) klasa velikih celih brojeva (*BigInt*). Potom napisati glavni program (kao konzolnu .exe aplikaciju) koji testira rad sa celim brojevima.

Specifikacija klase BigInt

Klasa `BigInt` predstavlja apstraktni tip podataka za rad sa označenim velikim celim brojevima. Veliki celi brojevi mogu imati neograničen broj cifara i sve navedene operacije se obavljaju u neograničenoj tačnosti. Za smeštanje cifara PREPORUČUJE se korišćenje kolekcija iz standardne biblioteke, ali se po želji mogu praviti i sopstvene strukture podataka. NIJE DOZVOLJENO koristiti biblioteke koje implementiraju slične ili iste funkcionalnosti!

Veliki broj se pravi na osnovu znakovnog niza cifara.
Za potrebe inicijalizacije, implementirati konstruktor `BigInt::BigInt(string digits);`. String `digits` mora sadržati isključivo cifre. Na prvom mestu, opciono se može pojaviti znak broja. Broj cifara je neograničen, pa se za smeštanje cifara MORA koristiti dinamička memorija.

Kopiranje velikog broja.
Implementirati obe varijante konstruktora kopije (duboko kopiranje i kopiranje sa premeštanjem) za potrebe kloniranja velikih brojeva.

Veliki brojevi moraju biti nepromenljivi (immutable).
Jednom kada se napravi objekat tipa BigInt i jednom kad mu se definiše vrednost, NE SME se više menjati. Zabraniti upotrebu operatora =. Sve operacije sa velikim brojevima prave nove privremene rezultate/objekte i NE SMEJU menjati vrednosti operanada.

Propisno uništavanje velikih brojeva.
Implementirati destruktor za propisno uništavanje objekata klase BigInt.

Sabiranje velikih brojeva.
Implementirati metodu `BigInt BigInt::add(const BigInt&) const;`.
Metoda sabira dva velika broja i vraća novi objekat klase BigInt. Voditi računa o znakovima operanada, kao i znaku rezultata.

Oduzimanje velikih brojeva.
Implementirati metodu `BigInt BigInt::sub(const BigInt&) const;`.
Metoda oduzima dva velika broja i vraća novi objekat klase BigInt. Voditi računa o znakovima operanada, kao i znaku rezultata.

Množenje velikih brojeva.
Implementirati metodu `BigInt BigInt::mul(const BigInt&) const;`.
Metoda množi dva velika broja i vraća novi objekat klase BigInt. Voditi računa o znakovima operanada, kao i znaku rezultata.

Deljenje velikih brojeva.
Implementirati metodu `BigInt BigInt::div(const BigInt&) const;`.
Metoda radi celobrojno deljenje dva velika broja i vraća novi objekat klase BigInt, pri čemu se ostatak zanemaruje. Voditi računa o znakovima operanada, kao i znaku rezultata.

Poređenje velikih brojeva (greater).
Implementirati metodu `bool BigInt::greaterThan(const BigInt& num) const;`.
Metoda vraća true ukoliko je vrednost broja za koji je pozvana veća od vrednosti broja num.

Poređenje velikih brojeva (less).
Implementirati metodu `bool BigInt::lessThan(const BigInt& num) const;`.
Metoda vraća true ukoliko je vrednost broja za koji je pozvana manja od vrednosti broja num.

Poređenje velikih brojeva (equals).

Implementirati metodu `bool BigInt::equals(const BigInt& num) const;`.
Metoda vraća true ukoliko je vrednost broja za koji je pozvana jednaka vrednosti broja num.

Ispisivanje velikih broja u izlazni tok.

Preklopiti operator << za ispisivanje objekta klase `BigInt` u izlazni tok, gde je prototip operacije `ostream& operator<<(ostream&, const BigInt&)`. Operatorska funkcija ispisuje sve cifre velikog broja uključujući i predznak, ako se radi o negativnom broju.

Test funkcija

U projektu glavnog programa treba da postoji funkcija `void test();` koja testira rad sa velikim brojevima. Studenti treba da implementiraju datu funkciju i uslovno je prevode ako je definisan makro STUDENT_TEST. Predvideti i postojanje makroa PROF_TEST. Ako su oba makroa definisana, PROF_TEST ima prioritet i tada se prevodi tajni test primer koji će biti dat na odbrani. Funkcija main treba samo da pozove test funkciju i na kraju ispiše njeno trajanje u milisekundama, korišćenjem tipova i operacija iz zaglavlja <ctime>.

Tehnički zahtevi i smernice za izradu rešenja

Sve klase i metode moraju biti imenovane prema zahtevima iz domaćeg zadatka. Poljima klase, zaštićenim od direktnog pristupa, se pristupa isključivo pomoću odgovarajućih metoda za čitanje i pisanje vrednosti polja. Za smešanje tekstualnih vrednosti upotrebiti tačno onoliko mesta u memoriji koliko je neophodno. Svaka klasa koja koristi dinamičku memoriju mora imati korektno napisan destruktor i konstruktor kopije. Izuzetno u ovom zadatku, klase sa destruktorom koji nije automatski generisan smeju imati automatski generisan operator dodele vrednosti, ali ga ne treba koristiti!

Glavni program treba da poziva metode/operacije koje obavljaju opisane radnje. Sve metode smestiti u odgovarajuće klase. Programski kod klasa rasporediti u odgovarajuće **.h** i **.cpp** fajlove. Nije dozvoljeno korišćenje globalnih promenljivih za razmenu podataka. Sva razmena podataka između funkcija mora ići preko povratne vrednosti i/ili liste argumenata. U slučaju bilo kakve greške (poziv programa sa neodgovarajućim brojem argumenata komandne linije, neuspešna dodela dinamičke memorije, greška pri radu sa datotekom ili bilo koja druga greška koja se može pojaviti u toku izvršavanja programa), ispisati odgovarajuću poruku i prekinuti izvršavanje.

Ukoliko u zadatku nešto nije dovoljno jasno definisano, treba usvojiti razumnu pretpostavku i na temeljima te pretpostavke nastaviti izgrađivanje svog rešenja.

VAŽNE NAPOMENE

Za uspešno odbranjen domaći zadatak potrebno je na odbrani pokazati kod podeljen na odgovarajuće projekte, .h i .cpp fajlove.

- Klase kojima su implementirani osnovni koncepti treba da budu smeštene u poseban projekat rešenja koji se prevodi kao statička biblioteka (**`bigint.lib`**).
- Glavni program napisati u posebnom projektu koji se prevodi kao Win32 Console Application (**`testdz1.exe`**) fajl i koji treba povezati sa statičkim bibliotekama. Glavni program treba implementirati tako da pozove globalnu funkciju `void test();`.
- Studenti treba da implementiraju svoju verziju ove funkcije tako da demonstriraju operacije sa velikim brojevima.
- Na kraju programa potrebno je ispisati na standardnom izlazu vreme trajanja funkcije `test` u milisekundama. Može se iskoristiti kod za merenje vremena na jeziku C++ koji je dat u zadatku 2.8 u materijalima za vežbe.
- NIJE DOZVOLJENO SMESTITI CEO KOD U JEDAN PROJEKAT ILI CPP fajl!

21.10.2015. godine sa predmeta