Domaća zadaća 1

Kalkulator

Zadatak je implementirati razred koji će u funkcionalnom smislu oponašati klasičan stolni kalkulator („digitron“). U zadatku ne morate implementirati nikakvo vizualno sučelje, već isključivo jedan razred koji možete nazvati po želji, no bitno je da bude u prostoru imena PrvaDomacaZadaca_Kalkulator i da implementira sljedeće sučelje:

```
public interface ICalculator
{
    void Press(char inPressedDigit);
    string GetCurrentDisplayState();
}
```

Funkcija Press() omogućuje unos znaka u kalkulator, a funkcija GetCurrentDisplayState() omogućuje uvid u trenutno stanje ekrana kalkulatora. Te dvije funkcije koristit će se u testovima. Kako bi se domaća zadaća mogla ocijeniti bez obzira na imenovanje i implementaciju samog razreda, za dohvat razreda koristit će se oblikovni obrazac tvornice. Zbog toga je potrebno implementirati razred Factory.

```
public class Factory
{
    public static ICalculator CreateCalculator()
    {
        // vratiti kalkulator
        return new <vlastiti kalkulator>();
    }
}
```

Rješenje je potrebno predati u obliku jedne datoteke, naziva po izboru, koja će sadržavati razred kalkulator i gore navedeni razred Factory. (Napomena: ne uključivati definiciju sučelja ICalculator u tu datoteku jer se zbog kolizije programski kôd neće moći prevesti.)

U nastavku je prikazan predložak izgleda datoteke koju je potrebno predati.

```
using …

namespace PrvaDomacaZadaca_Kalkulator
{
    public class OvoJeImeMogKalkulatora:ICalculator
    {
          … vaša implementacija kalkulatora
    }

    public class Factory
    {
         return new OvoJeImeMogKalkulatora();
    }
}
```

Kako bi vam se dodatno olakšalo rješavanje ovog zadatka, u repozitoriju datoteka, u mapi Domaće zadaće se pored teksta ovog zadatka nalazi i arhiva s Visual Studio solutionom, u kojem su već stavljena potrebna sučelja, a također imate i skup osnovnih unit-testova za provjeru koliko ste dobro riješili zadatak. Skup testova pokriva osnovnu funkcionalnost i postoje zahtjevi koji nisu pokriveni testovima koji se nalaze u primjerima (ali hoće biti pokriveno u širem skupu testova po kojem će se evaluirati vaša rješenja), tako da to svakako uzmete u obzir!

Rok za predaju rješenja je ponedjeljak, 18. studenoga 2019. u 8:00 sati.

Nažalost, Nemesis sustav za automatizirano ocjenjivanje zadaća više nije "među živima", tako da ćete svoja rješenja domaćih zadaća predati mailom, na sljedeći način.

Za predaju rješenja ćete napraviti direktorij s nazivom oblika Prezime_Ime, u taj direktorij ćete staviti svoju Kalkulator.cs datoteku, zatim ćete **cijeli direktorij** komprimirati u .zip arhivu (dobiti ćete datoteku oblika Prezime_Ime.zip) i tu arhivu mi poslati na mail: zvonimir.vanjak@gmail.com sa subjectom DZ1.

Napomena: s obzirom da ću ocjenjivanje rezultata testova pokrenutih nad vašim rješenjem obaviti korištenjem automatizirne skripte, **molim vas da se ovih uputa obavezno pridržavate kako ne bi ostali bez bodova!**

Važna napomena: Kod razvoja svog rješenja dopušteno je koristiti maksimalno .NET 3.5. (što znači da .NET 4.0 i .NET 4.5 NISU podržani!)

Zahtjevi na rješenje:

Ekran može maksimalno sadržavati **10 znamenki**, što znači da uz decimalni znak i predznak može sadržavati maksimalno **12.** U slučaju unosa više znamenki, potrebno je raditi samo s onim znamenkama koje stanu na ekran. Prilikom obavljanja operacija rezultat je potrebno **zaokruživati** u svakoj situaciju kad se sve znamenke rezultata ne mogu prikazati na „ekranu“ (odnosno, kad god je rezultat dulji od 10 znamenki, bilo cjelobrojnih bilo decimalnih!!! – dodatna napomena kako bi bilo potpuno jasno: kalkulator radi **isključivo s brojevima/operandima kako su mu prikazani na ekranu** i zanemaruje za daljnje operacije sve znamenke koje su otpale prilikom zaokruživanja!) .

Nakon „paljenja“ kalkulatora, na ekranu se prikazuje znamenka 0. Nakon svakog **brisanja** ekrana na ekranu se također prikazuje 0.

Kalkulator koristi **decimalni zarez**, a ne decimalnu točku. Decimalni zarez ispisuje se odmah na ekranu (znači, '2' ',' prikazuje „2,“). **PRI TOME OBAVEZNO PAZITI NA POSTAVLJENI "Culture"!!!**

Potrebno je podržati sljedeće matematičke operacije: zbrajanje, oduzimanje, množenje, dijeljenje, promjena predznaka, kvadriranje, korjenovanje, inverz, sinus, kosinus, tangens, spremanje broja u memoriju i dohvaćanje iz memorije. Svaka operacija mora biti predstavljena kraticom od jednog znaka, a popis kratica dan je u sljedećoj tablici. Za unarne operacije, prilikom pritiska na odgovarajuću tipku, operacija se obavlja nad trenutno prikazanim brojem na ekranu (dakle, za izračunati sin(5), najprije se unosi broj 5, a nakon toga se obavlja operacija pritiskom na tipku 'S' – u biti, to je standardni način rada kalkulatora).

Tablica 1 Popis operacija

Operacija	**Kratica**	**Opis kratice**
+	+	
-	-	
*	*	
/	/	
=	=	
Decimalni zarez	,	
Promjena predznaka	M	-naziv prema Minus
Sinus	S	-naziv prema Sinus
Kosinus	K	-naziv prema Kosinus
Tangens	T	-naziv prema Tangens
Kvadriranje	Q	-naziv prema Quadrat
Korjenovanje	R	-naziv prema Root
1/x	I	-naziv prema Invers
Spremanje u memoriju	P	-naziv prema Put
Dohvaćanje iz memorije	G	-naziv prema Get
Brisanje ekrana	C	-naziv prema Clear
Resetiranje kalkulatora	O	-naziv prema Off/On

Sve operacije istog su prioriteta te je potrebno omogućiti redanje operacija (na primjer, kod izračunavanja izraza "2+3-2^2*2", a koji se unosi pritiskom na slijedeći niz tipki: '2','+','3','-','2','Q','*','2' rezultat izračunavanja treba biti 2, a ne -3).

U slučaju pogreške prilikom obavljanje operacije potrebno je na ekran ispisati ''**-E-**''. Isto je potrebno ispisati i u slučaju da rezultat operacije ima više znamenki nego što je dopušteno (s obzirom da kalkulator NE podržava eksponencijalni prikaz za brojeve, ukoliko se cjelobrojni dio broja ne može prikazati treba ispisati ovu poruku o grešci).

Nakon obavljanja svake računske operacije (odnosno, nakon svakog pritiska na znak "=") broj na ekranu potrebno je zaokružiti s time da se eventualne nule viška također moraju maknuti (npr., ukoliko korisnik unese sljedeći niz znakova '2', ',', '0', '=' decimalni dio se nakon pritiska na '=' treba maknuti i na ekranu treba pisati samo „2“). Broj se zaokružuje i nakon binarnog operatora (npr. ukoliko korisnik unese sljedeći niz znakova '2', ',', '0', '+' decimalni dio se nakon pritiska na '+' treba maknuti i na ekranu treba pisati samo „2“).

Promjenu predznaka potrebno je implementirati da radi jednako kao i tipka "+/-" na tipičnom kalkulatoru, odnosno, pritisak na tu tipku mijenja predznak trenutno unesenog broja na ekranu (ukoliko nema nikakvog unesenog broja, ili se tek čeka njegov unos, tipka ne radi ništa!). Pretpostavlja se da trigonometrijske funkcije računaju s radijanima, a ne stupnjevima jer tako rade i funkcije iz biblioteke Math.

Radi jednostavnosti nije potrebno implementirati funkcije za rad s memorijom koje nude kalkulatori (M+, M-, MR, MC) već jednostavno spremanje i dohvaćanje iz memorije. U memoriju je moguće pohraniti samo jedan broj. Spremanjem broja u memoriju briše se prethodno spremljeni broj. Nakon spremanja broja u memoriju, korisniku je potrebno omogućiti unos drugih znamenki tog broja, tj. ne briše se ekran kalkulatora.

U slučaju brisanja ekrana ne brišu se podaci iz memorije niti rezultati prijašnjih operacija, a u slučaju gašenja i paljenja kalkulatora brišu se svi povezani podaci.

Potrebno je omogućiti skraćeni zapis operacija s istim operatorom (na primjer 2+= → 2+2= → 4).