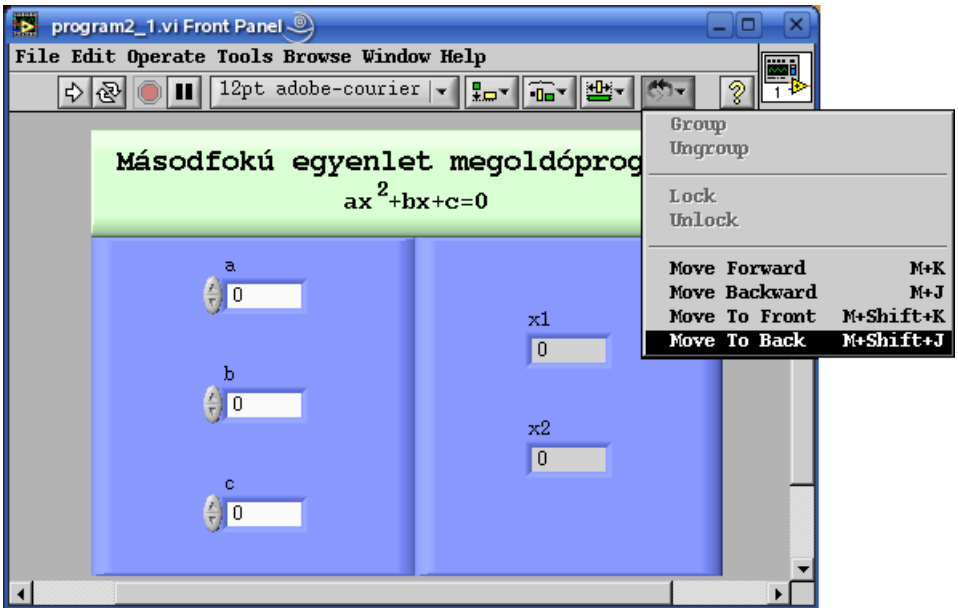
4. Példa: Másodfokú egyenlet megoldása (program2_1.vi)

Mielőtt nekilátnánk a programozásnak, idézzük fel a másodfokú egyenlet általános alakját, és ez alapján gondoljuk végig, hogy milyen elemekre lesz szükségünk a 'Front Panel' kialakításához.

$$ax^2+bx+c=0 \tag{1}$$

Az egyenlet három együtthatót és egy független változó tartalmaz. Tehát szükségünk lesz három bemeneti elemre és egy helyett kettő kijelzőre (hiszen az egyenletnek két gyöke lehet). Az elhelyezett elemek legyenek 'Digital Control' illetve 'Digital Indicator' típusúak. Ha elhelyeztük az elemeinket, akkor rendezzük el az előző fejezetben tanultak szerint. Az egyes csoportok (bemenetek – kijelzők) jobb elkülönítéséhez használjuk a 'Decorations' palettán lévő elemeket. Amennyiben egy ilyen dekorelem eltakarná a kijelzőinket, akkor a II.1. ábrán látható menü megfelelő pontjának kiválasztásával rendezhetjük a megfelelő mélységbe.

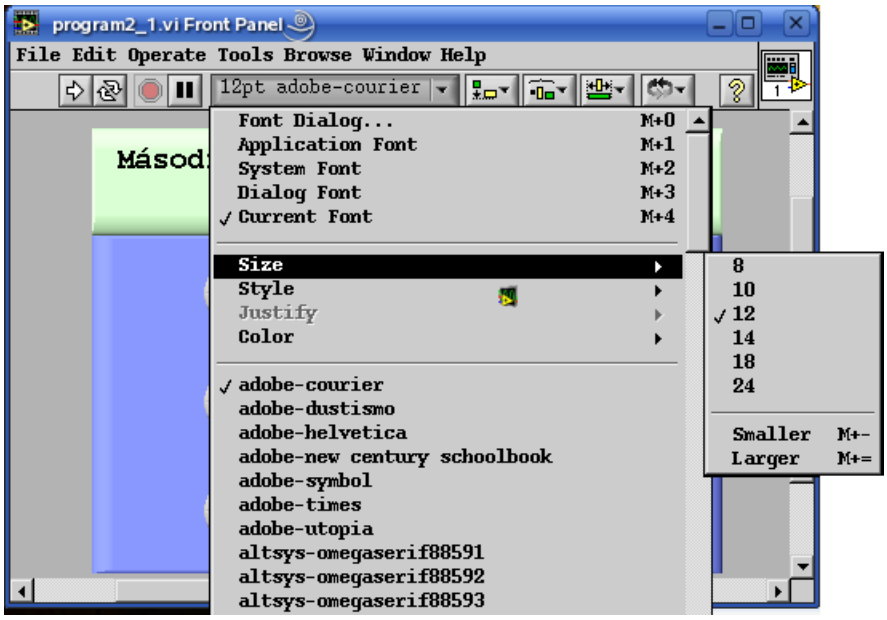


II.1. ábra: Egy lehetséges 'Front Panel' kialakítás

A 'Front Panel'-en és a 'Diagram Panel'-en is elhelyezhetünk un. független feliratokat. Amennyiben be van kapcsolva az helyzetérzékeny kurzor, akkor a panelek felületén egy dupla kattintást végezve egy szöveg beírására alkalmas elem keletkezik. A szöveg tulajdonságait (méret, stílus, igazítás) a középtájon lévő menüből állíthatjuk be (II.2. ábra). Az innen elérhető tulajdonságokon kívül nincs más formázási lehetőség. ezért például a hatványozás kettesét egy külön szövegmezőben lehet létrehozni és ezután a megfelelő pozícióba kell helyezni.
A menükben lévő 'M' betű a Meta billentyűre utal, aminek az IBM kompatibilis PC-k billentyűzetén az 'Alt' billentyű felel meg!
Ne felejtsük el beállítani a változóink 'Caption' értékét, valamint átállítani, hogy a 'Front Panel'-en ne a 'Label' tulajdonság, hanem ez kerüljön megjelenítésre!
Ha mindezzel készen vagyunk térjünk rá a tényleges programozási feladatra, mely a másodfokú egyenlet megoldóképletének (2) LabVIEW-beli megvalósítása.

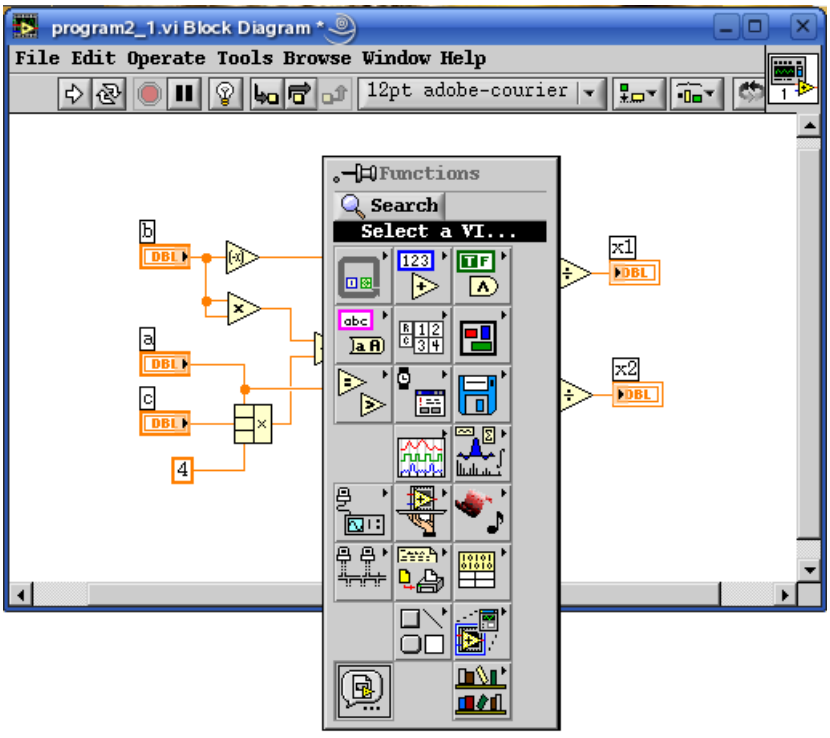
$$x_{1,2}=\frac{-b\pm\sqrt{b^2-4\cdot a\cdot c}}{2\cdot a} \tag{2}$$

Ehhez a feladathoz tulajdonképpen már minden ismerettel rendelkeznek, csupán a 'Numeric' palettáról kell a megfelelő elemeket kikeresni.



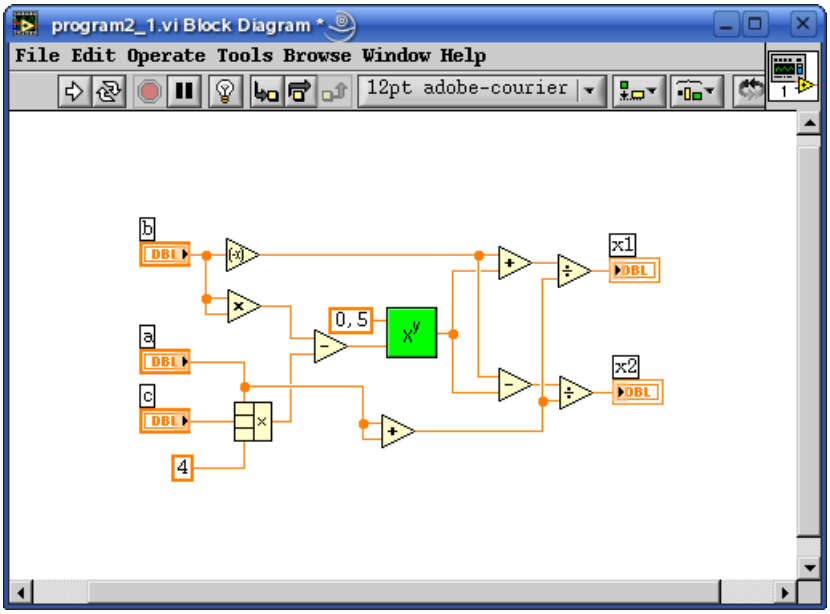
II.2. ábra: Betűk beállítása

Mivel mi nem csak ezeket az elemeket szeretnénk felhasználni, hanem a korábban elkészített hatványozónkat is fel akarjuk használni, ezért keressük meg a 'Functions' palettán a 'Select a VI' gombot (II.3. ábra)!



II.3. ábra: Saját készítésű 'SubVI' beillesztése

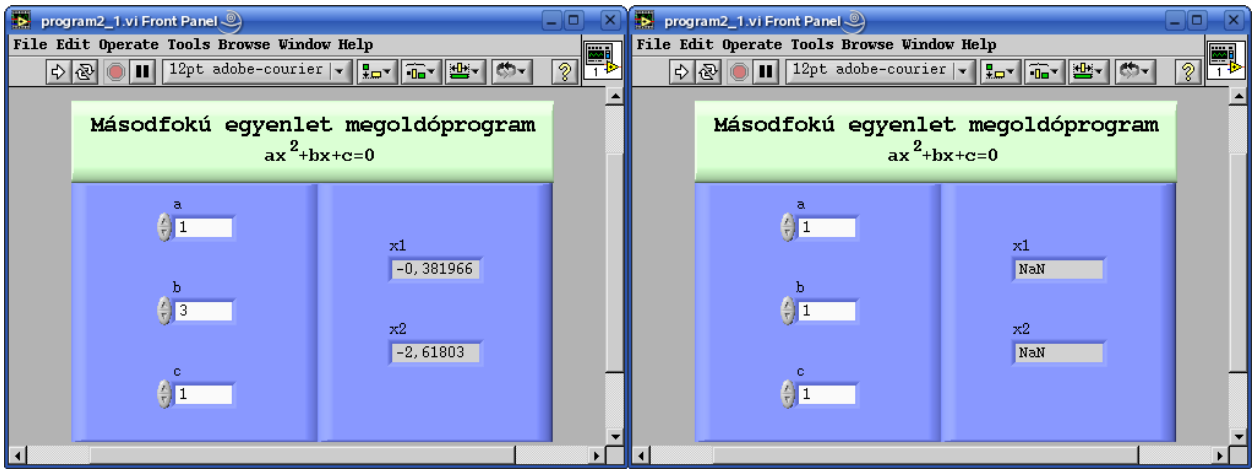
Az ekkor kinyíló fájlmegnyitás ablakban keressünk meg a hatványozó programunkat és helyezzük el a 'Diagram Panel'-en. Ha jól végeztük el a feladatot, akkor a II.4. ábrán láthatóhoz hasonló kell legyen a programunk.



II.4. ábra: A program 'Diagram Panel'-je

Egyetlen elem szorul némi magyarázatra, méghozzá a 'Compound Arithmetic' (), amely tulajdonképpen egy olyan elem, amely kettőnél több bemenet esetén képes elvégezni a következő műveleteket: összeadás, szorzás, logikai 'AND', 'OR' és 'XOR'. A műveletek között vagy az elem legördülő menüjén keresztül, vagy pedig az elemre működtető típusú kurzorral kattintva tudunk váltani (Figyeljük meg, hogy hasonlóképp működött a tartályos példában a 'Property Node' egy elemének kiválasztása is!). A kettőnél több bemenetet úgy tudjuk elérni, hogy ha az elem fölé visszük a kurzort, akkor az alsó és a felső részén egy-egy kis négyzet jelenik meg, amelyek segítségével elvégezhető az átméretezés (ez igaz az összes olyan LabVIEW-s elemre, amelynek nem csak meghatározott számú be- vagy kimenete lehet).

A program két futási eredményét láthatjuk a II.5. ábrán.



II.5. ábra: A program futtatásának eredménye

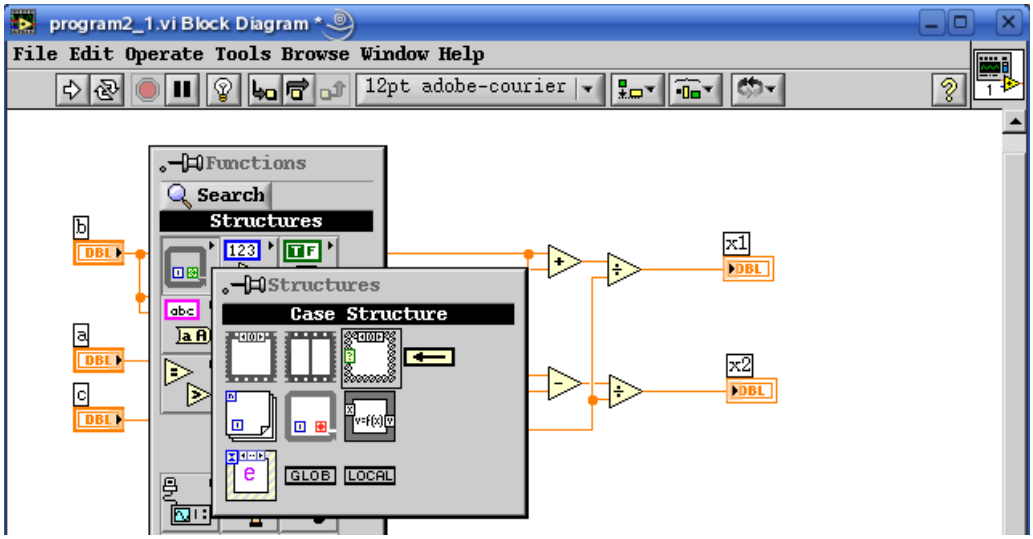
A jobb oldali panelen látható eredmény oka az, hogy az adott paraméterek mellett az egyenletnek komplex gyökei vannak, amit még nem tud kezelni a program. A következő táblázat azt mutatja, hogy milyen eseteket kell megvizsgálnunk, ha a programunkat minden eshetőségre fel akarjuk készíteni (a D betű a diszkriminánst jelöli).

<table>
<tr><td rowspan="2">a=0</td><td rowspan="2">b=0</td><td>c=0</td><td>Azonosság</td></tr>
<tr><td>c<>0</td><td>Butaság</td></tr>
<tr><td></td><td colspan="2">b<>0</td><td>Elsőfokú egyenlet</td></tr>
<tr><td rowspan="3">a<>0</td><td colspan="2">D=0</td><td>Kettős valós gyök</td></tr>
<tr><td colspan="2">D>0</td><td>Két valós gyök</td></tr>
<tr><td colspan="2">D<0</td><td>Komplex konjugált gyökpár</td></tr>
</table>

A táblázat segítségével fogalmazzuk meg, hogy melyek azok a funkciók, amelyeket szeretnénk a program végleges változatában benne látni.

- Minden eset kezelése.
- Egy tájékoztató mező, amely informál arról, hogy melyik eset áll fenn a hatból.
- Az egyes esetekben mindig a megfelelő számú kijelző megjelenítése.
- Az eredménymezők feliratai az adott esetnek megfelelő legyen.

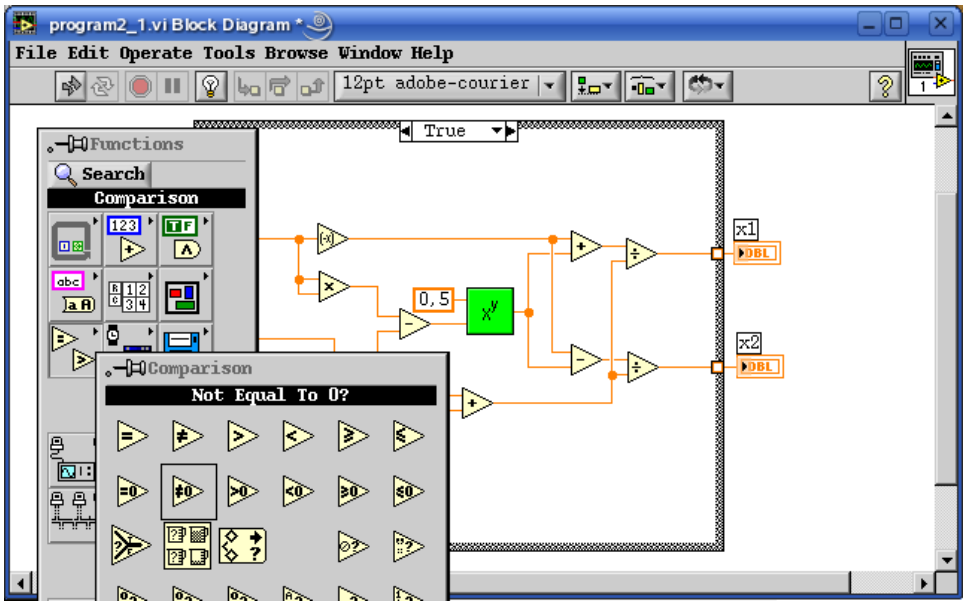
Az egyes esetek kezelésére a LabVIEW-ban a 'Case' struktúra használható, melyet természetesen a struktúrák alpalettán találhatunk meg (II.6. ábra). Elhelyezésekor a ciklusoknál tanult módon járjunk el.



II.6. ábra: A 'Case' struktúra

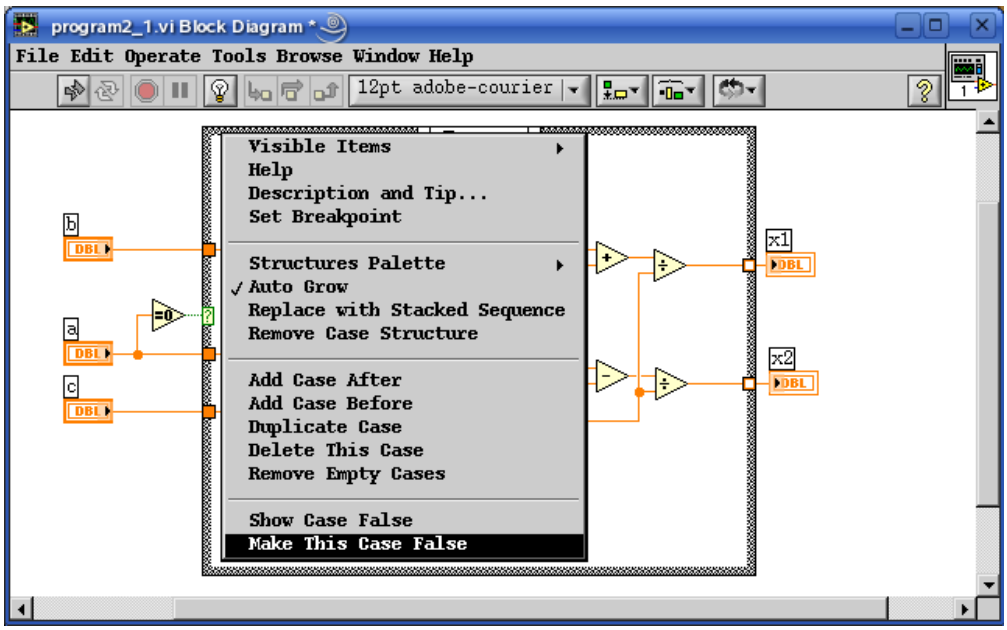
A 'Case' struktúra megfelel a szöveges programnyelvek ha-akkor-különben utasításinak, de megfelel a többszörös elágazásokat megvalósító utasássorozatoknak is. Hogyan is lehetséges ez? A 'Case' struktúrának van egy ún. 'Selector' bemenete (), melybe az elágazás feltételét kell bekötni. Ez lehet 'Boolean', 'Error Cluster', 'Integer' vagy 'String' típusú. Az első két típus esetén az egyszerű, míg a második két adattípus esetén a többszörös elágazást valósíthatunk meg vele. A 'Case' struktúrák tetszőleges mélységben helyezhetők el egymásban.

A programunkban először azt kell megvizsgálnunk, hogy az a=0 feltétel fenn áll-e. Ehhez a művelethez az összehasonlító függvényeket a 'Comparison' palettán (II.7. ábra) találjuk.



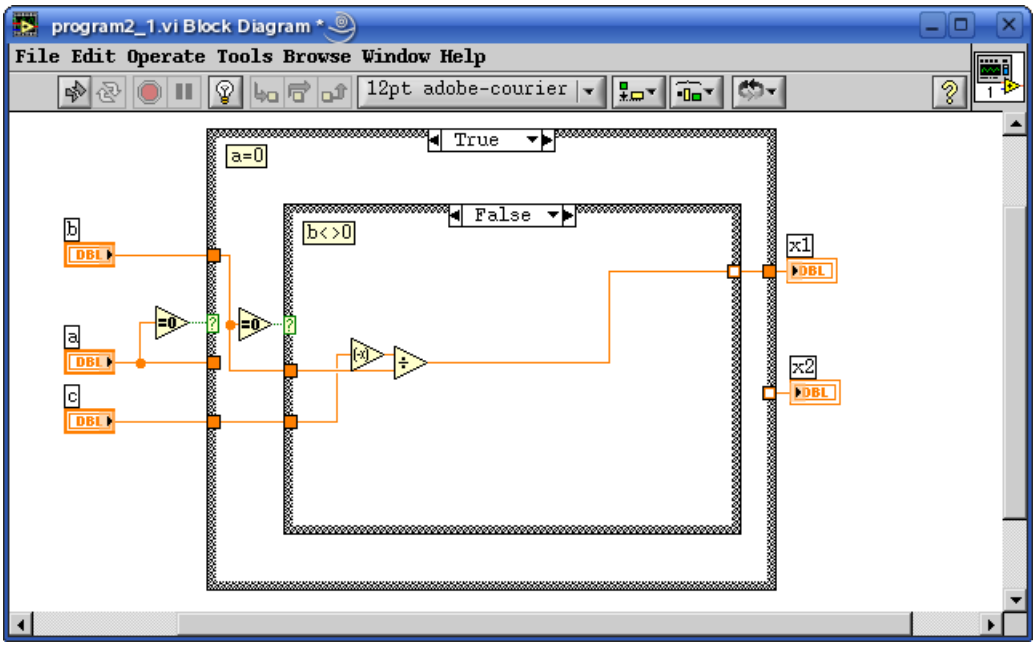
II.7. ábra: A 'Comparison' paletta

Válasszuk ki az 'Equal to 0' összehasonlító függvényt. Ha még nem tettük volna le a 'Case struktúrát, akkor helyezzük el úgy, hogy a kijelzők és kontrollok termináljain kívül minden a területére essen. A 'Case' struktúra fejlécét megnézve látható, hogy ez a programrészlet a 'True', vagyis az Igaz esethez fog tartozni, de mivel mi azt vizsgáljuk, hogy egyenlő-e az „a” változó nullával, és ez a programrészlet a két valós gyök esetén lesz érvényes, ezért át kell állítani, hogy ez a 'False' azaz Hamis esethez tartozzon. Ezt úgy tehetjük meg, hogy a struktúra keretére kattintunk az egér jobb gombjával, ez kiválasztjuk a 'Make This Case False' menüpontot (II.8. ábra).

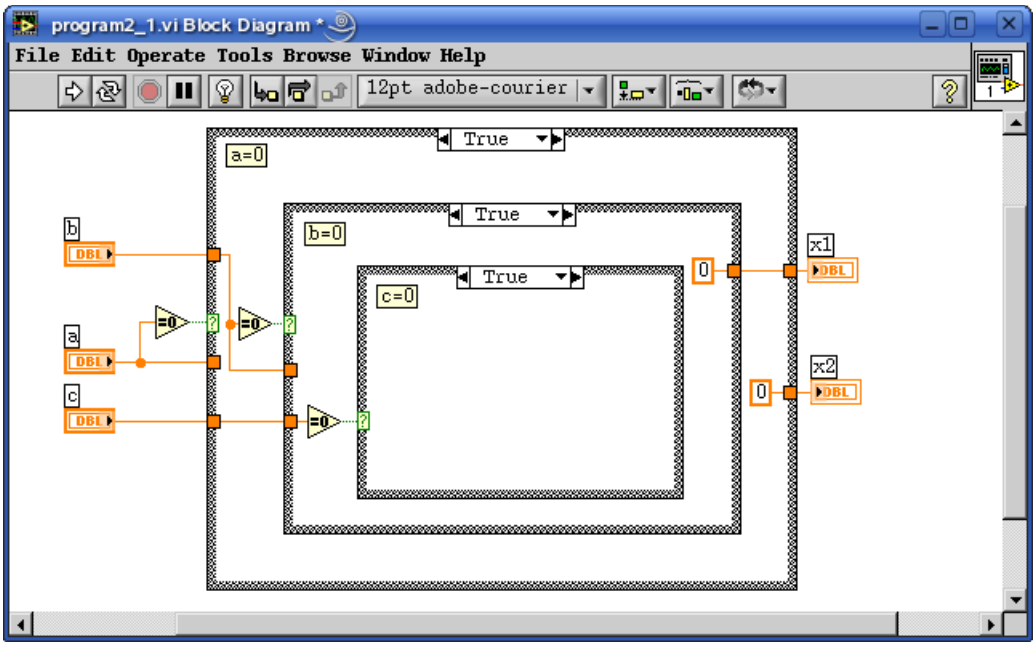


II.8. ábra: Az Igaz eset Hamissá tétele a 'Case' struktúrában

Kezdjük tehát az a=0 esetekkel. A következő 'Case' struktúránál azt kell figyelni, hogy 'b' értéke egyenlő-e nullával. Amennyiben nem, akkor egy elsőfokú egyenletünk van (II.9 ábra).
A számítások eredményét az 'x1' kijelzőbe kössük be! Figyeljük meg, hogy a külső elagázásnál az adatvonal kimeneti pontja a bekötés után telített lett az 'x1'-nél, míg 'x2'-nél, és a belső 'Case'-nél üres. Ha egy ilyen 'Tunnel' telített, az azt jelenti, hogy minden esetben van hozzá rendelve egy kimeneti érték. Amennyiben üres, akkor valamelyik esetnél ez hiányzik, és ez a programban hibaként jelentkezik. Tehát a 'Case' struktúránál, minden egyes belőle kifelé haladó adatvonalnál az összes esetben adatot kell rendeljünk.

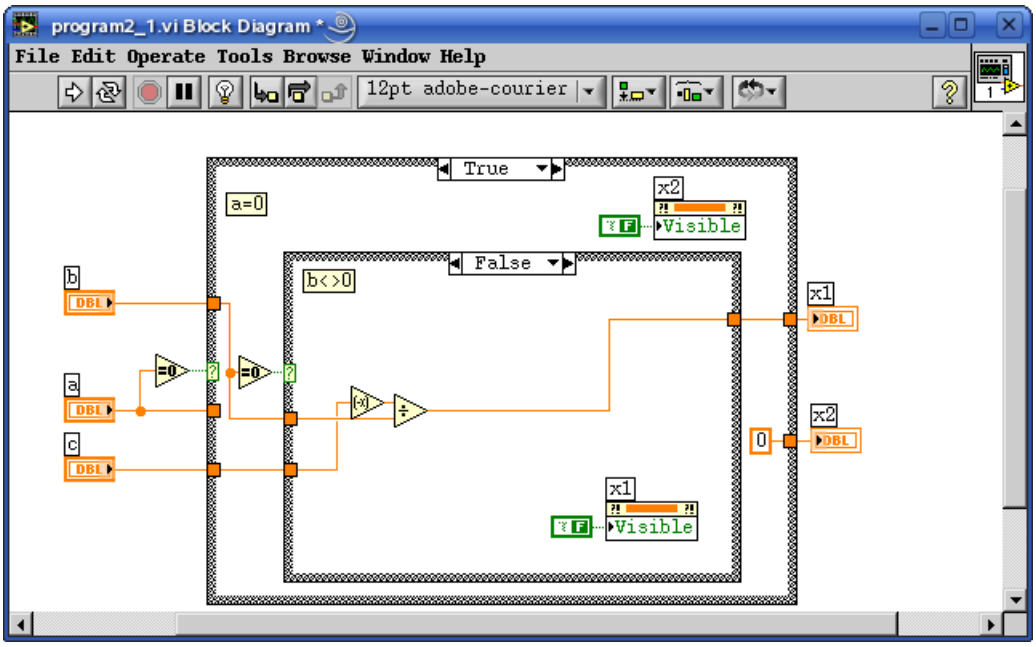


II.9. ábra: Az elsőfokú egyenlet esete



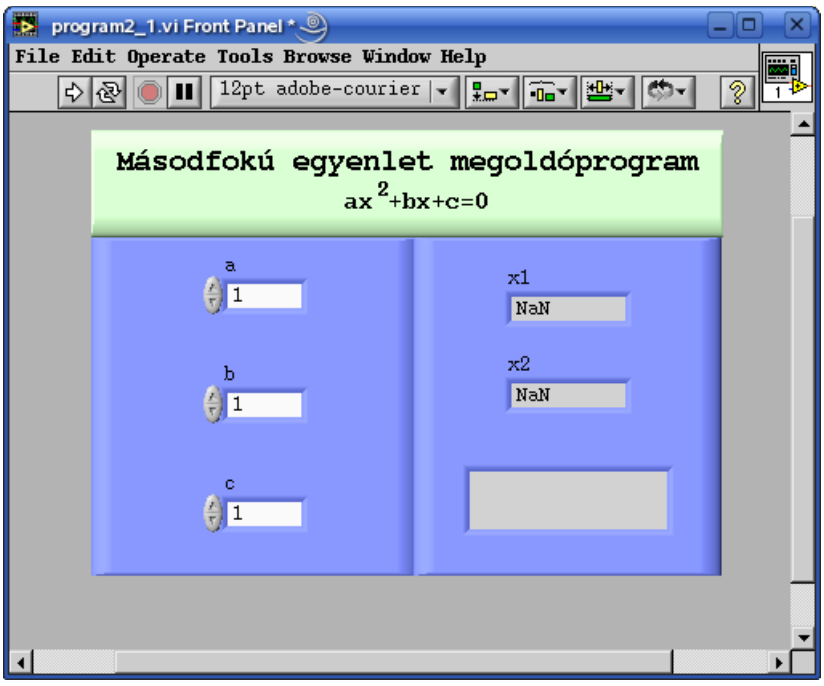
II.10. ábra: Az első három eset struktúrái

Ez a magyarázata annak, hogy a II.10. ábrán miért kötöttünk be zérus értékeket kimenetként, holott azoknál az eseteknél tulajdonképpen érdektelen illetve nem használt az adott kijelző.



II.11. ábra: A nem használt kijelző(k) elrejtése

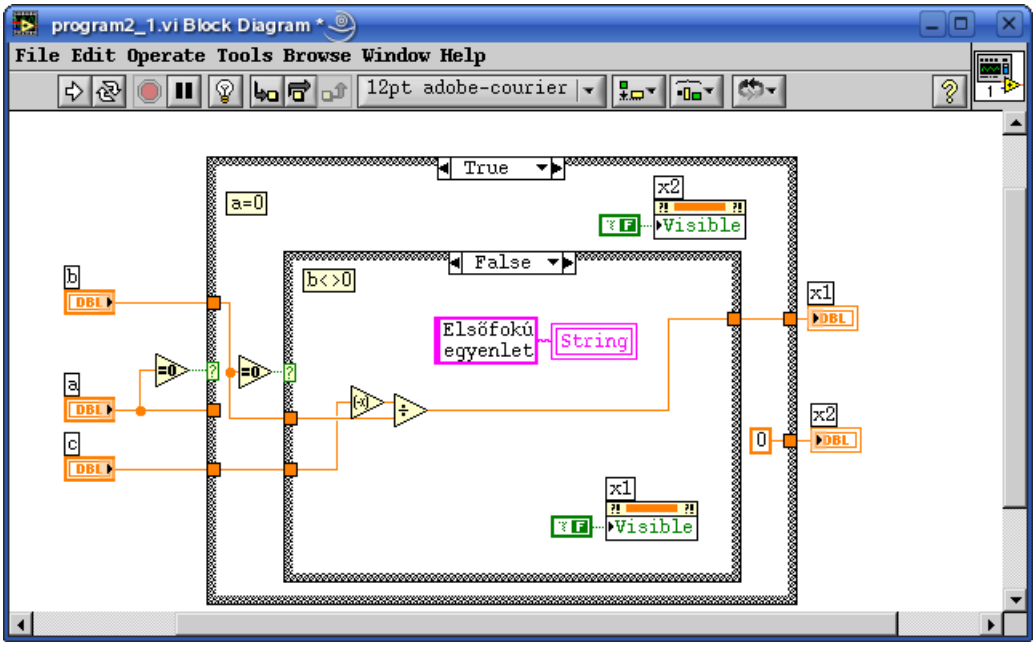
A nem használt kijelzőt el is rejthetjük. Ehhez már ismerjük a az objektumok tulajdonságainak futásközbeni állíthatóságát lehetővé tevő 'Property Node'-ot, melynél ezúttal a 'Visible' tulajdonságot fogjuk a helyzetnek megfelelően beállítani (II.11. ábra).



II.12. ábra: A végleges 'Front Panel'

Ahhoz, hogy a különböző esetekhez a tartozó tájékoztatást meg tudjuk adni a felhasználónak,

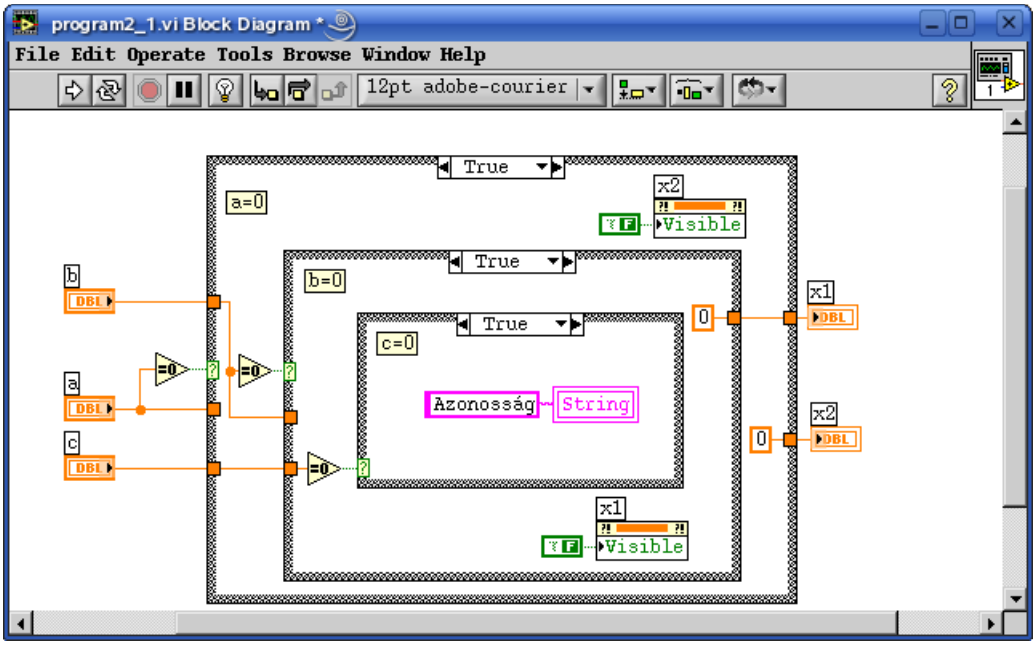
helyezzünk el a 'Front Panel'-en egy 'String Indicator' mezőt. Ez a kijelző, hacsak nem állítjuk be, hogy egy soros legyen, alapértelmezettet egy több soros szöveg megjelenítésére alkalmas, de hogy lássuk is a sorokat célszerű a függőleges méretét megnövelni.



II.13. ábra: Üzenet kiíratása

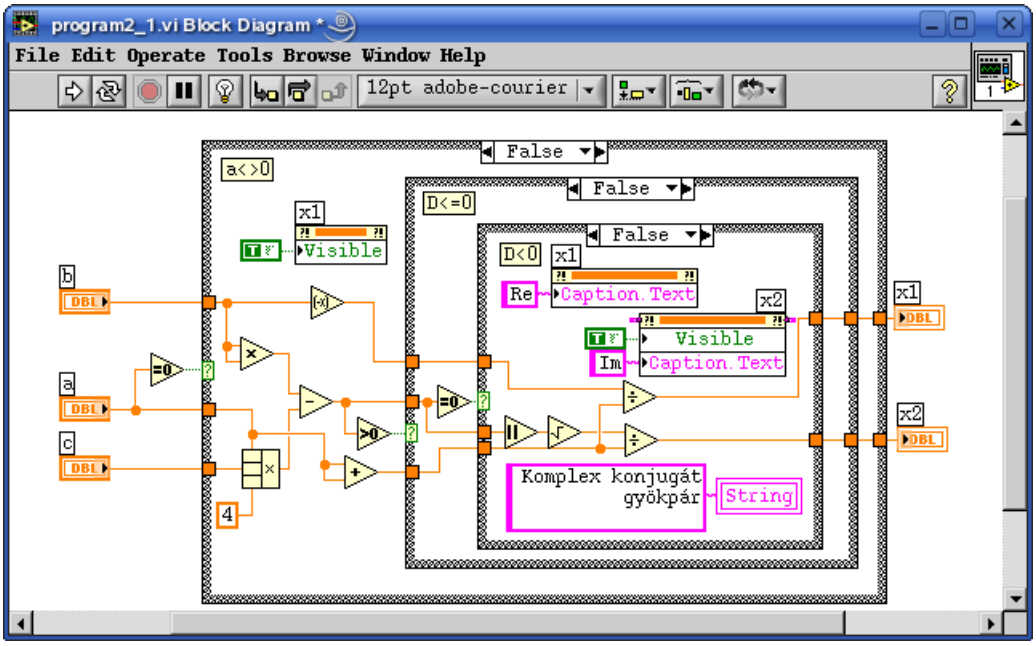
Most pedig ismerkedjünk meg a lokális változókkal. Lokális változót két módon hozhatunk létre. Az első lehetőség, hogy a 'Functions' paletta struktúrák alpalettájáról vesszük le a 'Local Variable'-t. Ekkor a lehelyezett elem fekete és egy kérdőjelet tartalmaz, ami azt jelenti, hogy még nincs hozzárendelve egy változóhoz sem. Ezt a hozzá tartozó legördülő menün keresztül tehetjük meg. A másik, némileg kényelmesebb út, hogy annak a változónak amelyhez a lokális változót el akarjuk készíteni, a legördülő menüjéből kiválasztjuk a 'Create'->'Local Variable' alpontot, és ekkor egy már változóhoz rendelt lokális változót hoz létre a LabVIEW. A lokális változók, csakúgy mint a „rendes" változók, vagy írhatóak, vagy olvashatóak. DE, például egy kijelző típusú elemnek, amely csak írható, a lokális változója lehet akár írható, de lehet olvasható is! Ez a lokális változók alkalmazásának legnagyobb előnye, de legnagyobb veszélye is (hogy miért, arra még a későbbiekben visszatérünk).
A felhasználó tájékoztatására szolgáló szöveges mezőt tehát lokális változók segítségével fogjuk az adott esetnek megfelelő szöveggel ellátni. Ebből kifolyólag megtehetnénk azt is, hogy nem rendelünk minden esetben értéket hozzá, hiszen nem kell egy adatvonalon illetve 'Tunel'-en keresztül kivinni az adatokat a 'Case' struktúrából.
Ezek alapján készítsük el az a=0 esetekhez tartozó programrészeket, majd térjünk át az $a \neq 0$ esek tárgyalására. Itt már egy esetre, a valós gyökök esetére, vagyis ha a diszkrimináns nagyobb zérusnál, elkészült a programunk. Ezért vizsgáljuk ilyen módon a lehetséges variációkat.
A másik esetet tovább kell bontanunk aszerint, hogy a diszkrimináns egyenlő-e zérussal (II.16 .ábra), vagy kisebb (II.15. ábra) nála.
Amennyiben a kisebb, akkor az eredmény egy komplex konjugált gyökpár lesz, amelyikből mi csak a valós illetve a komplex részüket szeretnénk kiíratni. De ekkor az eddigi „x1" illetve „x2" feliratoknak nem lesz értelme, hiszen inkább „Re" illetve „Im" feliratokat kellene az egyes kijelzőkre kitenni. Ezt a feladatot szintén a 'Property Node'-dal tudjuk elvégezni. Vagy készítünk a kijelzőinkhez újabb 'Property Node'-okat, vagy pedig – mivel átméretezhető elemek – széthúzzuk a meglévőket és a második mezőben a nekünk most szükséges 'Caption.Text'

tulajdonságot állítjuk be.



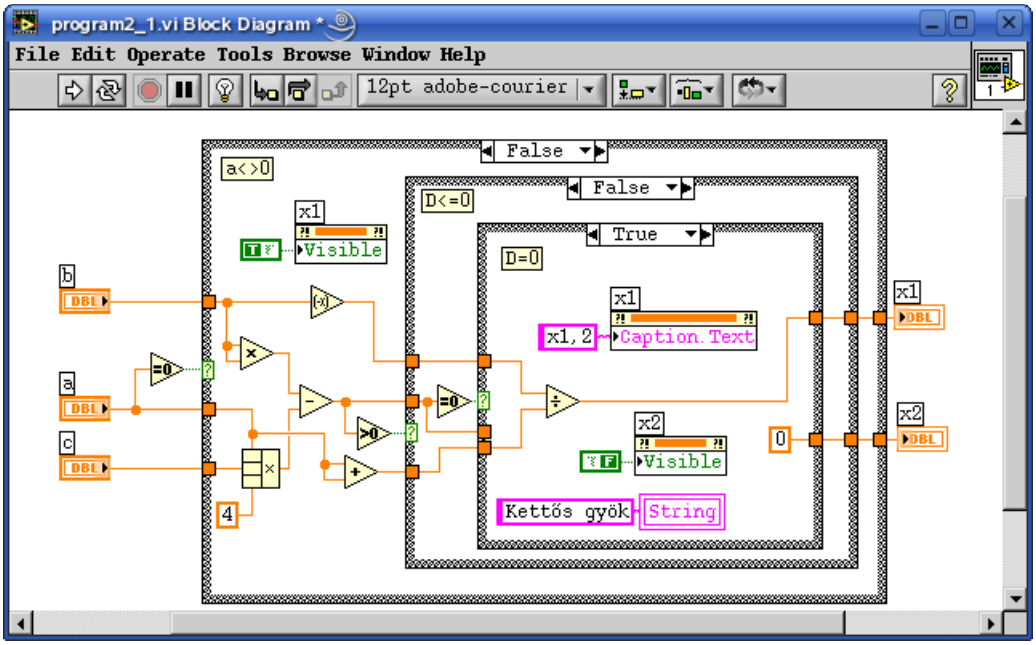
II.14. ábra: Az azonosság esete

Hasonló módon az összes esetnél be kell állítanunk a kijelző feliratát, kivéve ahol az egész kijelző láthatóságát letiltottuk. Ne felejtsük el az elsőfokú egyenlet esetében is beállítani az egyes kijelző 'Caption'-jét „x”-re.!

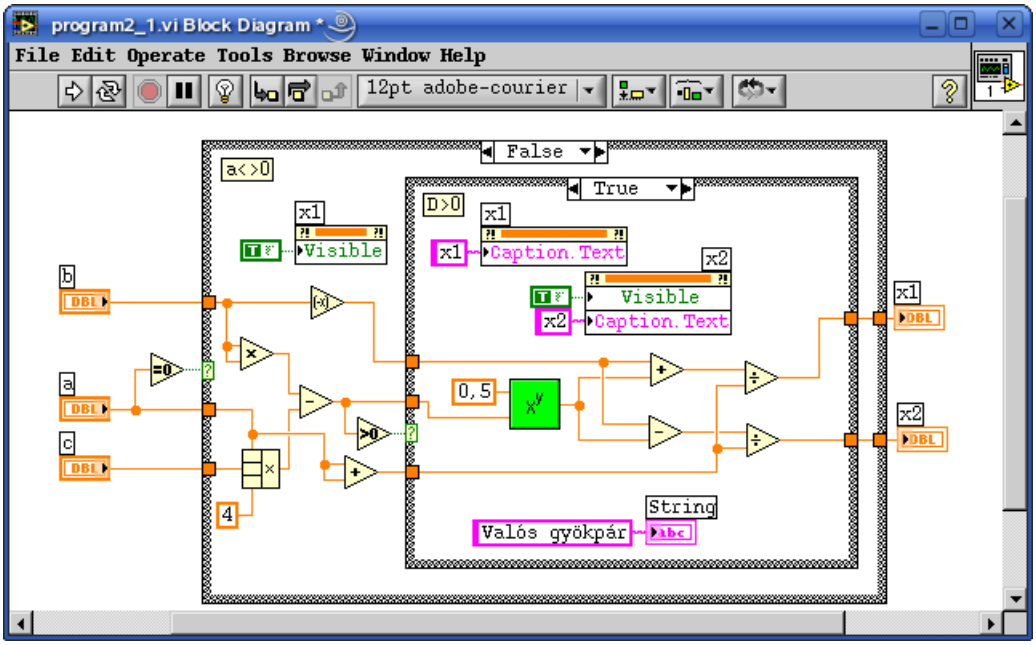


II.15. ábra: A komplex konjugált gyökpárok esete

A másodfokú egyenlet különböző megoldásainak 'Block Diagram'-jai a II.15, II.16 valamint a II.17 ábrákon találhatók.



II.16. ábra: A kettős gyök esete



II.17. ábra: A két valós gyök esete