Psimulex teendők

Fontossági sorrend szerint (magas, közepes, alacsony)

# Core-beli bővítések:

* Típusrendszer befejezése (PriorityQueue, Tree, Graph egyszerű típusok)
* Rekord típus kezelése
* A programban zajló események kivezetése (ValueChanged, stb.)
* Párhuzamossághoz szükséges elemek (legalább egy lock)
* A Library-k és típusok exportfelületének testreszabhatósága
* Include kezelése (ez közös a fordítóval)
* Visszatekerés a programban
* Memóriahasználat mérése
* Statisztikai modulhoz kellő információk biztosítása (elég, hogyha korrekt eseménykezelés megoldható)
* Delay funkció befejezése (az utasítások hatásának késleltetése)
* A különböző léptetési módok (StepIn, StepOut, StepOver, Run) elkészítése
* Extra típusok felvétele (bináris fa, kupac, stb.)
* Breakpointok kezelése, highlight kijelölés meghatározása (nem biztos, hogy Core feature)
* A virtuális gépek állapotainak mentése és visszatöltése (sorosítás)
* Immediate Window funkcióhoz támogatás a Core-ban (Execute(string command))
* CPU Cache Kozsik javaslatára.
* A futásidejű információk bővítése (melyik fájl melyik sorában, melyik utasításnál vagyunk pontosan). Hiba vagy „kivétel” dobásakor az egész CallStack-et kiírása.
* PVM szimuláció

# User Interface (alapfelület):

* Felület megtervezése és létrehozása (fájl tallózó, forrásfájlok megnyitása külön fülekben, a megjelenítési vezérlők dokkolhatósága)
  + A felületről testreszabni a virtuális gépeket, megmondani, hogy melyiken épp mi fusson.
  + A debuggolás a virtuális gép egy folyamatához kötődik. A megjelenítés szálakra vonatkozik, tehát a szálaknak külön grafikus ábrázolásuk lesz. Erre figyelni kell, hiszen szálanként kell tudni esetleg konfigurálni a megjelenést, láthatóságot...
* Objektumok ábrázolása
  + Típusok megjelenítése (skaláris és gyűjtemények, hierarchikus és gráfszerű adatszerkezetek).
  + Az előzőre épülően valamennyi típusra megfelelő input ablakok létrehozása
  + A megjelenítőben lehessen megváltoztatni az objektum értékét és szerkeszteni a megjelenését
* Include-olt fájlok okos kezelése (ezt tervezni is kell még)
* Az aktuális állapot mentése (gépek és a megjelenítés)
* A futtatási ablakban az értékek reprezentációja testreszabható grafikus elemek legyenek, azaz, lehessen őket elrejteni, mozgatni, átméretezni és megjelenésileg testreszabni (színezés, Show As... funkció, amikor másmilyen típusként próbálja megjeleníteni). Itt lennének elérhetőek pl. a gráfelrendezések is.
* Hatékonyság-elemzés.
* Breakpoint, Run to Cursor kezelése és megjelenítése.
* StepBack funkció. A felületen egy ablakban a program teljes története megtekinthető volna, amik között egy kattintással lehetne ugrálni. Az első futás után a felvett állapotváltozások alapján visszajátszható és oda- visszatekercselhető a program futása.
* Különféle inputok implementálása (fájl, string, excel, stb.)
* Core kiegészítése grafikus Library-vel. Lehessen kódból elrendezni a grafikus elemeket pl. egy gráf esetében, színezni, stb.
* A változók elrejtésének szintaxfára vonatkozó beállítása, diffelés a változtatás követésére
* Immediate Window
* Intellisense (típuskövetkeztetés, ahol lehet. Tree.Value-nél nem kell)
* Adatszerkezet demó

# Psimulex.DemoDesigner (ez a demó tervezőnk):

* A Core és a UserInterface felhasználásával lehessen magától futó (gombnyomásig megálló) lejátszást tervezni bizonyos programoknak.
* Az adatok bekérését a demó leírójában lehet kihagyásra kényszeríteni, hogyha beégetünk előre értékeket (pl. stringként). De ha interakció is fontos, akkor meg lehet hagyni a bekérő utasításokat, hogy felhozzák a megfelelő formokat.
* Folytatni a demó tervező megtervezését, mert még csak ötletek vannak.