

Zestaw 6 – Functional Global Variable (FGV)

Poniższe zadania realizuj jako *Functional Global Variable*. Każdy FGV twórz w oknie projektu.

Zad. 6.1

Zbuduj FGV „Counter”. Funkcja powinna na wejściu posiadać wartość boolowską „Reset(T)/Count(F)”. W stanie Count, powinna ona zliczać swoje kolejne wywołania. Stan Reset powinien pozwalać na zresetowanie licznika.

Zad. 6.2

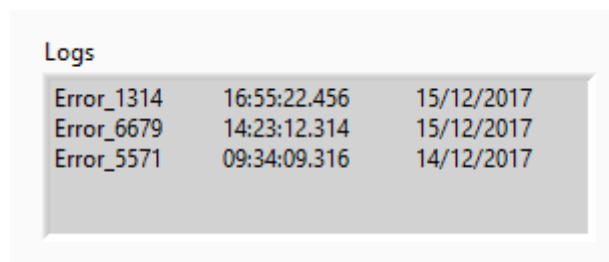
Napisz FGV „Timer”. Program powinien posiadać jedno wejście typu enum o nazwie *Action* (elementy: Reset, Run, Pause) i jedno wyjście numeryczne typu DBL o nazwie *Time Elapsed [ms]*. Uruchomienie programu z wartością *Action* ustawioną na Reset powoduje, że na wyjściu V1a otrzymujemy zero. Każde kolejne uruchomienie programu z wartością *Action* ustawioną na Run zwraca wartość w milisekundach informującą, ile czasu upłynęło od ostatniego resetu. Wykorzystanie *Action* Pause wstrzymuje naliczanie czasu, aż do zmiany wartości *Action* na Reset lub Run.

Zad. 6.3

Zbuduj FGV „Logs”. Funkcja ta powinna służyć jako baza danych dla logów z programu. Powinna przechowywać ostatnie zapisane do niej logi i zwracać je w postaci stringa z kolejnymi rekordami ułożonymi w kolejności od najświeższych rekordów (tj. od góry), a z najstarszymi u dołu.

Funkcja na wejście powinna przyjmować:

- enum informujący o typie wykonywanej akcji (ClearList, AddRecord, GetList)
- cluster zawierający informację (string) oraz czas i datę rejestracji loga (timestamp)
- ilość rekordów gromadzonych w historii. Po przekroczeniu zadanej liczby, najstarsze rekordy są usuwane.



Error_1314	16:55:22.456	15/12/2017
Error_6679	14:23:12.314	15/12/2017
Error_5571	09:34:09.316	14/12/2017

Zad. 6.4

Napisz FGV „Database”, działający jako baza danych użytkowników. Podprogram powinien mieć opcję dodania nowego użytkownika, zwracania listy użytkowników przy wcześniejszym posortowaniu listy użytkowników po nazwisku oraz czyszczenia listy użytkowników.

Dane użytkownika do przechowywania: Imię, Nazwisko, Wiek, Narodowość