```
using TypeBindingSample;
using Microsoft.Modeling;

config Main
{
    action all AccountImpl;
    switch StateBound = 12800;
    switch StepBound = 12800;
    switch PathDepthBound = 12800;
    switch TestClassBase = "vs";
    switch GeneratedTestPath = "..\\TestSuite";
    switch GeneratedTestNamespace = "TypeBindingSample";
    switch TestEnabled = false;
}

// We do not need to indicate the domain for account because Spec Explorer will use all
// instances created during the whole exploration as the domain
// DomainGenerator is not supported for object instances
config ParameterCombinationConfig : Main
{
    action static void AccountImpl.SetBalance(Account account, float balance)
    where {. Condition.In(balance, 10, 50); .};  //EITHER 10 OR 50

    action static Set<Account> AccountImpl.SearchAccounts(float balance)
    where {. Condition.In(balance, 20, 100); .};
}

machine ModelProgram() : Main   //exploration will fail as it is too unconstrained
{
    construct model program from ParameterCombinationConfig
}

machine SlicedModelProgram() : Main
{
    (CreateAccount;(SetBalance|GetBalance|SearchAccounts)*; Clear) || ModelProgram
}

machine jp1() : Main
{
    (CreateAccount; CreateAccount; CreateAccount; CreateAccount;(SetBalance)*) || ModelProgram
}

machine jp2() : Main
{
    (CreateAccount; CreateAccount; (SetBalance | GetBalance)*; Clear) || ModelProgram
}

machine TestSliced() : Main where TestEnabled = true
{
    construct test cases for SlicedModelProgram
}

machine Testjp1() : Main where TestEnabled = true
{
    construct test cases for jp1
}

machine Testjp2() : Main where TestEnabled = true
{
    construct test cases for jp2
}
```