Exemplul cu tati redefinit

Exercitiu:

Sa se afiseze tatii, folosind ca si clause persoana si parinte.
main.pro

```
implement main
    open core

    domains
        gen = fata(); baiat().

    class facts - fapteFamilia
        persoana : (string Nume, gen Gen).
        parinte : (string Persoana, string Copil).

    class predicates
        tata : (string Tata, string Copil) nondeterm anyflow.
    clauses
        tata(Tata, Copil) :-  parinte(Tata, Copil) , persoana(Tata, baiat).

    clauses
        run():-
            console::init(),
            file::consult("..\\familia.txt", fapteFamilia),

            stdIO::write("Lista tatilor:\n"),
            tata(Tata, Copil),
                stdIO::write(Tata, " este tatal lui ",  Copil , "\n"),
            fail.

        run():-
            stdIO::write("-------------------------\n"),
            stdIO::write("Tatal lui Tudor este "),
            tata(Tata, "Tudor"),
                stdIO::write(Tata),
            fail.

        run():-
            stdIO::write("\nSfarsitul testarii\n").
end implement main
```

```
goal
   mainExe::run(main::run).
```

familia.txt

```
clauses
   persoana("Gabriel" , baiat).
   persoana("Ion" , baiat).
   persoana("Adriana" , fata).
   persoana("Monica" , fata).
   persoana("George", baiat).

   parinte("Gabriel" , "Sorin").
   parinte("Ion" ,  "Dorina").
   parinte("Adriana" , "Tudor").
   parinte("Monica" , "Sandu").
   parinte("George" , "Tudor").
```

Rularea:

```
Lista tatilor:
Gabriel este tatal lui Sorin
Ion este tatal lui Dorina
George este tatal lui Tudor
----------------------------
Tatal lui Tudor este George
Sfarsitul testarii
```

Tati si mame

Afisati tatii si mamele, folosind ca si clause persoana si parinte.

main.pro

```
implement main
   open core

   domains
      gen = fata(); baiat().

   class facts - fapteFamilia
      persoana : (string Nume, gen Gen).
      parinte : (string Persoana, string Copil).

   class predicates
      tata : (string Tata, string Copil) nondeterm anyflow.
   clauses
      tata(Tata, Copil) :-  parinte(Tata, Copil) , persoana(Tata, baiat).

   class predicates
```

```
    mama : (string Mama, string Copil) nondeterm anyflow.
clauses
    mama(Mama, Copil) :-  parinte(Mama, Copil) , persoana(Mama, fata).

clauses
    run():-
        console::init(),
        file::consult("..\\familia.txt", fapteFamilia),

        stdIO::write("Lista tatilor:\n"),
        tata(Tata, Copil),
            stdIO::write(Tata, " este tatal lui ",  Copil , "\n"),
        fail.

    run():-
    stdIO::write("--------------------------\n"),
        stdIO::write("Lista mamelor:\n"),
        mama(Mama, Copil),
            stdIO::write(Mama, " este mama lui ",  Copil , "\n"),
        fail.

    run():-
        stdIO::write("\nSfarsitul testarii\n").
end implement main

goal
    mainExe::run(main::run).
```

familia.txt

```
clauses
    persoana("Gabriel" , baiat).
    persoana("Ion" , baiat).
    persoana("Adriana" , fata).
    persoana("Monica" , fata).
    persoana("George", baiat).
    parinte("Gabriel" , "Sorin").
    parinte("Ion" ,  "Dorina").
    parinte("Adriana" , "Tudor").
    parinte("Monica" , "Sandu").
    parinte("George" , "Tudor").
```

Rulare:

```
Lista tatilor:
Gabriel este tatal lui Sorin
Ion este tatal lui Dorina
George este tatal lui Tudor
--------------------------
Lista mamelor:
Adriana este mama lui Tudor
Monica este mama lui Sandu

Sfarsitul testarii
```

Tati, mame si bunici

Afisati tatii, mamele si bunicii, folosind ca si clause persoana si parinte.

main.pro

```
implement main
    open core

    domains
        gen = fata(); baiat().

    class facts - fapteFamilia
        persoana : (string Nume, gen Gen).
        parinte : (string Persoana, string Copil).

    class predicates
        tata : (string Tata, string Copil) nondeterm anyflow.
    clauses
        tata(Tata, Copil) :-  parinte(Tata, Copil) , persoana(Tata, baiat).

    class predicates
        mama : (string Mama, string Copil) nondeterm anyflow.
    clauses
        mama(Mama, Copil) :-  parinte(Mama, Copil) , persoana(Mama, fata).

   class predicates
        bunic : (string Bunic, string Nepot) nondeterm anyflow.
    clauses
        bunic(Bunic, Nepot) :-
  parinte(Bunic, Parinte) , parinte(Parinte, Nepot), persoana(Bunic,baiat).

 class predicates
        bunica : (string Bunica, string Nepot) nondeterm anyflow.
    clauses
        bunica(Bunica, Nepot) :-
  parinte(Bunica, Parinte) , parinte(Parinte, Nepot), persoana(Bunica,fata).

    clauses
        run():-
            console::init(),
            file::consult("..\\familia.txt", fapteFamilia),
```

```
        stdIO::write("Lista bunicilor:\n"),
        bunic(Bunic, Nepot),
            stdIO::write(Bunic, " este bunicul lui ",  Nepot , "\n"),
        fail.

    run():-
        bunica(Bunica, Nepot),
            stdIO::write(Bunica, " este bunica lui ",  Nepot , "\n"),
        fail.

     run():-
        stdIO::write("\nSfarsitul testarii\n").
end implement main

goal
   mainExe::run(main::run).
```

familia.txt

```
clauses
   persoana("Gabriel" , baiat).
   persoana("Ion" , baiat).
   persoana("Adriana" , fata).
   persoana("Monica" , fata).
   persoana("George", baiat).

   parinte("Gabriel" , "Sorin").
   parinte("Ion" ,  "Dorina").
   parinte("Adriana" , "Tudor").
   parinte("Monica" , "Sandu").
   parinte("George" , "Tudor").
   parinte("Sorin" ,  "Andrei").
   parinte("Adriana" , "Petre").
   parinte("Tudor" , "Dorel").
```

```
Lista bunicilor:
Gabriel este bunicul lui Andrei
George este bunicul lui Dorel
Adriana este bunica lui Dorel

Sfarsitul testarii
```

Rulare:

Stramosi

Afisati tatii, mamele, bunicii si stramosii, folosind ca si clause persoana si parinte.

main.pro

```
implement main
    open core

    domains
        gen = fata(); baiat().

    class facts - fapteFamilia
        persoana : (string Nume, gen Gen).
        parinte : (string Persoana, string Copil).

    class predicates
        tata : (string Tata, string Copil) nondeterm anyflow.
    clauses
        tata(Tata, Copil) :-  parinte(Tata, Copil) , persoana(Tata, baiat).

    class predicates
        mama : (string Mama, string Copil) nondeterm anyflow.
    clauses
        mama(Mama, Copil) :-  parinte(Mama, Copil) , persoana(Mama, fata).

    class predicates
        bunic : (string Bunic, string Nepot) nondeterm anyflow.
    clauses
        bunic(Bunic, Nepot) :-
    parinte(Bunic, Parinte) , parinte(Parinte, Nepot), persoana(Bunic,baiat).

    class predicates
        bunica : (string Bunica, string Nepot) nondeterm anyflow.
    clauses
        bunica(Bunica, Nepot) :-
    parinte(Bunica, Parinte) , parinte(Parinte, Nepot), persoana(Bunica,fata).

    class predicates
        stramos : (string Stramos, string Persoana) nondeterm anyflow.
    clauses
        stramos(Stramos, Persoana) :-  parinte(Stramos , Persoana).
        stramos(Stramos, Persoana) :-
    parinte(Stramos , AltaPersoana), stramos(AltaPersoana,Persoana).

%     alta forma de scriere
```

```
%     stramos(Stramos, Persoana) :-
%             parinte(Stramos , Persoana);
%             parinte(Stramos , AltaPersoana),
%           stramos(AltaPersoana,Persoana).

        clauses
            run():-
                console::init(),
                file::consult("..\\familia.txt", fapteFamilia),

                stdIO::write("Lista stramosi:\n"),
                stramos(Stramos, Persoana),
                    stdIO::write(Stramos, " este stramosul lui ",  Persoana , "\n"),
                fail.

            run():-
                stdIO::write("\nSfarsitul testarii\n").
    end implement main

    goal
        mainExe::run(main::run).
```

familia.txt

```
clauses
    persoana("Gabriel" , baiat).
    persoana("Ion" , baiat).
    persoana("Adriana" , fata).
    persoana("Monica" , fata).
    persoana("George", baiat).
    parinte("Gabriel" , "Sorin").
    parinte("Ion" ,  "Dorina").
    parinte("Adriana" , "Tudor").
    parinte("Monica" , "Sandu").
    parinte("George" , "Tudor").
    parinte("Sorin" ,  "Andrei").
    parinte("Adriana" , "Petre").
    parinte("Tudor" , "Dorel").
```

Rulare:

```
Lista stramosi:
Gabriel este stramosul lui Sorin
Ion este stramosul lui Dorina
Adriana este stramosul lui Tudor
Monica este stramosul lui Sandu
George este stramosul lui Tudor
Sorin este stramosul lui Andrei
Adriana este stramosul lui Petre
Tudor este stramosul lui Dorel
Gabriel este stramosul lui Andrei
Adriana este stramosul lui Dorel
George este stramosul lui Dorel

Sfarsitul testarii
```