```
%Name Abrham Tamiru
% Class (CECS 342-07)
% Project Name (Prog 3 – language history(prolog))
% Due Date (10/17/2023)
% I certify that this program is my own original work. I did not copy any
part of this program from
% any other source. I further certify that I typed each and every line of
code in this program.



language(fortranl,1956).
language(lispl,1958).
language(algol60,1960).
language(cobol,1960).
language(pll,1964).
language(smalltakl,1968).
language(pascal,1970).
language(prolog,1970).
language(scheme,1974).
language(ckr,1978).
language(ml,1976).
language(fortran77,1977).
language(smalltalk80,1980).
language(ada83,1983).
language(commonlisp,1984).
language(sml,1984).
language(cplus,1984).
language(perl,1986).
language(eiffel,1986).
language(caml,1986).
language(tcl,1988).
language(python,1990).
language(fortran90,1990).
language(java,1994).
language(javascript,1994).
language(ruby,1994).
language(perl5,1994).
language(ocaml,1996).
language(schemer5rs,1998).
language(haskell98,1998).
language(cplusiso,1998).
language(csharp,2000).
language(python2,2000).
language(java5,2004).
language(csharp2,2004).
language(kotlin,2010).
language(go,2008).
language(rust,2010).
language(haskell2010,2010).
language(csharp5, 2012).
language(java8,2014).
language(swift,2014).
language(javascriptES2017,2017).
```

```
parent(fortranl,algol60).
parent(fortranl, pll).
parent(fortranl, fortran77).
parent(cobol, pll).
parent(algol60, pll).
parent(algol60, smalltalk).
parent(algol60, scheme).
parent(algol60, perl).
parent(algol60, ckr).
parent(algol60, cplus).
parent(algol60, tcl).
parent(lisp, scheme).
parent(lisp, commonLisp).
parent(pll, pascal).
parent(smalltalk, smalltalk80).
parent(pascal, ada83).
parent(pascal, python).
parent(scheme, commonLisp).
parent(scheme, schemer5rs).
parent(fortran77, fortran90).
parent(ml, sml).
parent(ml, haskell98).
parent(ckr, python).
parent(ckr, cplus).
parent(smalltalk80, ruby).
parent(smalltalk80, go).
parent(ada83, eiffel).
parent(sml, caml).
parent(cplus, python).
parent(cplus, java).
parent(cplus, cplusISO).
parent(cplus, rust).
parent(perl, perl5).
parent(perl, ruby).
parent(perl, perl5).
parent(eiffel, ruby).
parent(caml,ocaml).
parent(python, ruby).
parent(python, python2).
parent(java, javascript).
parent(java, java5).
parent(java, csharp).
parent(cplusiso, csharp).
parent(javascript, kotlin).
parent(javascript, javascriptES2017).
parent(javascript, kotlin).
parent(ruby, swift).
parent(csharp, csharp2).
parent(python2, go).
parent(python2, swift).
parent(schemer5rs, rust).
parent(haskell98, haskell2010).
parent(ocaml, rust).
```

```
parent(java5, kotlin).
parent(java5, java_8).
parent(csharp2, kotlin).
parent(csharp2, csharp5).
parent(csharp2, rust).
parent(csharp5, swift).
parent(csharp5, javascriptES2017).
parent(rust, swift).
parent(haskell2010, swift).

ancestor(X,Y):-parent(X,Y).
ancestor(X,Y):-parent(Z,Y),ancestor(X,Z).

descendant(X,Y):-ancestor(Y,X).

sibiling(X,Y):-parent(Z,X),parent(Z,Y), X\=Y.

sibiling_of_cplus(X):-sibiling(X,cplus).

cousin(X,Y):- ancestor(Z,X), ancestor(Z,Y), X\=Y.

ancestor_of_cplus(XL):-setof(X,ancestor(X,cplus),XL).
cousin_of_cplus(XL):-setof(X,cousin(X,cplus),XL).

related(rust,commomlist):- ancestor(rust,commonlist).
related(rust,commomlist):- descendant(commonlist,rust).
swift_desendant_lips(swift,lips).
older_than_2000(Language) :- language(Language, Year), Year < 2000.
```