LABORATORIO #1. PRIMEROS EJERCICIOS SOBRE LISTAS

-Integrantes:

- Pedro Luis Caricari Torrejón (Grupo 12)
- Leonardo Henry Añez Vladimirovna (Grupo 12)
- Oliva Rojas Gerson(Grupo 12)
- Vidal Cespedes Erick Edwing(Grupo 12)

-Porcentaje Completado: 100%

-Comentario(s):

-En la experiencia de hoy hemos aprendido a usar las listas, y hemos visto que las listas en prolog se tratan de dos maneras como cabeza y cola de lista, claramente esto esta hecho para ser tratado de forma recursiva.

Aunque hemos tenido ciertos problemas para poder realizar algunos ejercicios, fueron por sintaxis de código y supimos arreglarlo investigando y debugeando.

-Codigo Fuente:

```
% 1. Menor
menor([X],X).
menor([X|L],Menor):- menor(L,Menor1),
                          menor(X,Menor1,Menor).

menor(A,B,A):- A<B,!.
menor(_,B,B).

% 3. Se encuentra
seEncuentra([X|_],X).
seEncuentra([X|L],X):-iguales([X,L]).
% 5. Suma
suma([],0).
suma([X|L],Suma):- suma(L,Suma1),
                        Suma is Suma1+X.
% 7. Ordenado
ordenado(L):- ordenadoAsc(L);ordenadoDesc(L).

ordenadoAsc([]) .
ordenadoAsc([_]) .
ordenadoAsc([X,Y|L]):- X =< Y ,ordenadoAsc([Y|L]).

ordenadoDesc([]) .
ordenadoDesc([_]) .
ordenadoDesc([X,Y|L]):- X >= Y ,ordenadoDesc([Y|L]).

% 9. Iguales
iguales([]).
iguales([_]).
iguales([X,X|L]):-iguales([X|L]).

% 11. puroPares
puroPares([]).
puroPares([X]):- par(X).
puroPares([X|L]):-  par(X),
                          puroPares(L).
```

```
par(X):- Mod is (X mod 2),Mod=:=0.
impar(X):- not(par(X)).

% 13. puroImpares
puroImpares([]).
puroImpares([X]):- impar(X).
puroImpares([X|L]):- impar(X),
                          puroImpares(L).

% 15. existeImpar
existeImpar(X):- not(puroPares(X)).

% 17. ascendenteK
ascendenteK([],_) .
ascendenteK([_],_) .
ascendenteK([X,Y|L],K):- (X+K) =:= Y, ascendenteK([Y|L],K).

% 19. disjuntos
disjuntos(L1,L2):-
not(subconjunto(L1,L2)),not(subconjunto(L2,L1)).

pertenece([X|_],X).
pertenece([_|L1],Y):- pertenece(L1,Y).

subconjunto([],_).
subconjunto([X|L1],L2):- pertenece(X,L2),
                 subconjunto(L1,L2).

%iguales
iguales([]).
iguales([_]).
iguales([X,X|L1]):- iguales([X|L1]).

%2 mayor
may(A,B,A):- A>B, !.
may(_,B,B).

mayor([X],X).
mayor([X|L1], M):- mayor(L1,M1),
             may(M1,X,M).

%4 subconjunto
pertenece([X|_],X).
pertenece([_|L1],Y):- pertenece(L1,Y).

subconjunto([],_).
subconjunto([X|L1],L2):- pertenece(X,L2),
                 subconjunto(L1,L2).
```

```
%6 cantidad
cant([],0).
cant([_|L1],C):- cant(L1, C1),
           C is C1+1.
%8 frecuencia
igual(A,A).
frec([],_,0).
frec([X|L1],E,C):- frec(L1,E,C1),
                igual(X,E),
                C is C1+1.
%10 diferentes
dif([]).
dif([_]).
dif([X|L1]):- frec(L1, X, C),
              C =:= 1,
              dif(L1).

%12 puro primo
primo(N):- primo(N,2).
primo(N,I):- I > N//2, !.
primo(N,I):- N mod I =\= 0,
          I1 is I+1,
          primo(N,I1).

puroprim([]).
puroprim([X|L1]):- primo(X),
                puroprim(L1).

%14 existePar
existepar([X]):- X mod 2 =:= 0.
existepar([X|L1]):- X mod 2 =:= 0;
                    existepar(L1).

%16 existeParImpar
existeimpar([X]):- X mod 2 =:= 1.
existeimpar([X|L1]):- X mod 2 =:= 0;
                      existeimpar(L1).

existepi(L1):-existepar(L1),
              existeimpar(L1).

%18 mismosEle
mismosEle(L1, L2):- subconjunto(L1,L2),
                    subconjunto(L2,L1).
%20 imparPar
ser(A,B):- A mod 2 =:= 0,
           B mod 2 =:= 1.
imparPar([]).
imparPar([_]).
imparPar([A,B|L1]):- ser(A,B),
                  imparPar(L1).
```