```
(*Practica 1: Andreu Mut Portes. Ejercicios*)

(*Ejercicio 1*)

Ex1[cad_, a_] := Module[{},
                  módulo
  Return[Length[Position[cad, a]]]
  retorna   longitud   posición
 ]

Ex1[{a, b, a, b, a}, a]

3

Ex1[{a, a, a, b, a, b, a, b, b, b, b}, b]

3

6

Ex1[{a, a, a, b, a, b, a, b, b, b, b}, b]

6
```

```
(*Ejercicio 2*)

Ex2[x_, n_] := Module[{res, i},
               módulo
  res = {};
  (*Bucle*)
  For[i = 1, i ≤ n, i++,
  para cada
   res = Join[res, x];
         junta
  ];
  Return [res]
  retorna
 ]

Ex2[{a, b}, 3]

{a, b, a, b, a, b}
```

```
(*Ejercicio 3 Prefixos*)
```

```
Ex3[x_] := Module[{res, i, w},
           módulo
  res = {{}};
  (*Bucle amb Take i Append*)
              toma    añade
  For[ i = 1, i ≤ Length[x], i++,
  para cada           longitud
   w = Take[x, {1, i}];
       toma
   AppendTo[res, w];
   añade al final
  ];
  Return [res]
  retorna
 ]

Ex3[{a, a, b, a, b}]
```

```
{{}, {a}, {a, a}, {a, a, b}, {a, a, b, a}, {a, a, b, a, b}}
```

```
(*Ejercicio 4 Sufixos*)

Ex4[x_] := Module[{res, i, w},
           módulo
  res = {{}};
  (*Bucle amb Take i Append*)
              toma    añade
  For[ i = 1, i ≤ Length[x] , i++,
  para cada           longitud
   w = Take[x, {i, Length[x]}];
       toma           longitud
   AppendTo[res, w];
   añade al final
  ];
  Return [res]
  retorna
 ]

Ex4[{a, a, b, a, b}]
```

```
{{}, {a, a, b, a, b}, {a, b, a, b}, {b, a, b}, {a, b}, {b}}
```

```
(*Ejercicio 5 Segments*)
```

```
Ex5[x_] := Module[{res, i, w, j},
           módulo
  res = {{}};
  (*Bucle amb Take i Append*)
               toma    añade
  For[ i = 1, i ≤ Length[x], i++,
  para cada            longitud
       For[ j = i, j ≤ Length[x], j++,
       para cada           longitud
            w = Take[x, {i, j}];
                toma
            AppendTo[res, w];
            añade al final
         ];
  ];
  Return [res]
  retorna
 ]

Ex5[{a, a, b}]
```

```
{{}, {a}, {a, a}, {a, a, b}, {a}, {a, b}, {b}}
```

```
(*Ejercicio 6 For per a L1 (i) j per a L2. Join de la iesima de L1
              para cada                          junta
 amb jesima de L2. AppentTo amb la cadena resultant al resultat.*)

Ex6[x_List, y_List] := Module[{res, i, w, j},
                       módulo
  res = {};
  (*Bucle amb Take i Append*)
               toma    añade
  For[ i = 1, i ≤ Length[x], i++,
  para cada            longitud
       For[ j = 1, j ≤ Length[y], j++,
       para cada           longitud


            w = Join [x[[i]], y[[j]]];
                junta
            AppendTo[res, w];
            añade al final
         ];
  ];
  Return [Union[res]] (*El union e gasta per a eliminar repetits*)
  retorna   unión
 ]

l1 = {{}, {a}, {b}, {a, a, b}, {b, b, b, b}};

l2 = {{}, {a, a}, {b, b}};
```

```
Ex6[l1, l2]
```

```
{{}, {a}, {b}, {a, a}, {b, b}, {a, a, a}, {a, a, b}, {a, b, b}, {b, a, a}, {b, b, b},
 {b, b, b, b}, {a, a, b, a, a}, {a, a, b, b, b}, {b, b, b, b, a, a}, {b, b, b, b, b, b}}
```

```
(*Ejercicio 7 Union.*)
```

unión

```
Ex7[x_, y_] := Module[{res},
```

módulo

```
  res = {};
  res = Union[x, y];
```

unión

```
  Return [res];
```

retorna

```
 ]

l1 = {{a}, {b, b}, {a, b, a}};

l2 = {{}, {b, b}, {b, b, a}};

Ex7[l1, l3]
```

```
{{}, {a}, {b, b}, {a, b, a}, {b, b, a}}
```

```
(*Ejercicio 8 Utilitzar el modul del exercici 6*)

Ex8[x_List, n_Integer] := Module[{res, i, w, aux},
```

entrada módulo

```
  res = x;
  For[i = 1, i < n, i++,
```

para cada

```
      res = Ex6[x, res];
  ];

  Return [res];
```

retorna

```
 ]

l1 = {{a}, {b, b}, {a, b, a}};

l2 = {{a}, {b}};

Ex8[l2, 3]
```

```
{{a, a, a}, {a, a, b}, {a, b, a}, {a, b, b}, {b, a, a}, {b, a, b}, {b, b, a}, {b, b, b}}
```

```
(*Ejercicio 9*)
```

```
Ex9[x_List] := Module[{i, contA, contB, res},
                módulo
  contA = 0;
  contB = 0;
  For[i = 1, i ≤ Length[x], i++,
  para cada          longitud
      If[x[[i]] == a, contA++];
      si
      If[x[[i]] == b, contB++];
      si
  ];
  res = False;
      falso
  If[Mod[contA, 2] == 0 && contB > 2, res = True];
  si  operación módulo                  verdadero

  Return[res];
  retorna
 ]

l9 = {a, b, a, b, a, b, a};

Ex9[l9]
```

```
True
```

```
(*Exercici 10*)

Ex10[x_List] := Module[{contB, i, aux, res},
                módulo
  contB = 0;
  aux = Reverse[x];
      invierte orden
  For[i = 1, i ≤ Length[aux], i++,
  para cada          longitud
      If[aux[[i]] == b, contB++];
      si
      If[aux[[i]] == c, i = Length[aux] + 1];
      si                longitud
   (* Trencar el bucle quan te trobes la primera C*)
                                                  cons

  ];
  res = False;
      falso
  If[Mod[contB, 2] == 0, res = True];
  si  operación módulo          verdadero
  Return[res];
  retorna
 ]
```

```
l10 = {a, b, c, b, b, b, b};

Ex10[l10]
```

```
True
```

```
(*Exercici 11*)

Ex11[L1_List, L2_List] := Module[{posL2, i},
                          módulo
  posL2 = 1; (*<- Recorre les posicions de la llista curta.*)
      For[i = 1, i ≤ Length[L1], i++,
      para cada          longitud
            If[L1[[i]] == L2[[posL2]],
            si
    (* Si coincideixen les lletres, passa a la seguent possicio de la L2.*)
             posL2++;
             ];

            If[posL2 > Length[L2], (*Si en algun moment la posicio se n'ix
            si             longitud
     de L2 es que ja l'ha plenat del tot i la paraula estara dins*)
                  Return [True];
                  retorna  verdadero
             ];
       ];
      Return[False]; (*Si no se aplega a completar la paraula,
      retorna  falso
  torna False perque no estara.*)
        falso
 ]

l1 = {a, b, a, a, b, a, a, b, a};

l2 = {b, b, b, a, a};

Ex11[l1, l2]
```

```
False
```

```
(*Exercici 12. Utilitzar Take[l1, {n1,n2}]. n2= 1 + Length[n2] - 1*)
                           toma                          longitud
```

```
Ex12[L1_List, L2_List] := Module[{aux, i},
                          módulo
  For[i = 1, i ≤ Length[L2] - (Length[L1] + 1), i++,
  para cada          longitud          longitud
   aux = Take[L2, {i, i + (Length[L1] - 1)}];
         toma              longitud
   If[aux == L1,
   si
        Return[i];
        retorna
   ];
  ];
  Return[False];
  ret···  falso
 ]

l1 = {b, a, a, b};

l2 = {b, a, b, a, b, b, b, a, b, a, a, b, b, b, b, a, b, a, b, b, a, a, b, a};

l3 = {a, a, a, a};

Ex12[l1, l2]

9

Ex12[l3, l2]

False
```

```
(*Exercici 13:
 El Ex13A es una modificacio del 12 per a que torne totes les posicions,
no nomes la primera. El Ex13 es el que torna el que toca*)

Ex13A[L1_List, L2_List] := Module[{w, i, j, res},
                           módulo
      res = {};
      For[i = 1, i ≤ Length[L2] - (Length[L1] + 1), i++,
      para cada          longitud          longitud
       w = Take[L2, {i, i + (Length[L1] - 1)}];
           toma              longitud
       If[w == L1,
       si
            AppendTo[res, i]
            añade al final
        ];
       ];
  Return[res];
  retorna
 ]
```

```
Ex13[L1_List, L2_List] := Module[{res, aux, i},
                          módulo
      res = {};
      For[i = 1, i ≤ Length[L1], i++,
      para cada          longitud
            aux = Ex13A[L1[[i]], L2];
       If[aux ≠ {},
       si
             AppendTo[res, {aux, L1[[i]]}]
             añade al final
        ];
       ];
  Return[res];
  retorna
 ]

l1 = {{b, b}, {a, b, b, b}, {b, b, a, b}, {a, a, a, a}};

l2 = {b, a, b, a, a, b, b, a, b, b, b,
   a, b, b, a, b, a, a, a, a, a, b, b, a, a, b, b, a, b, a};

Ex13[l1, l2]

{{{6, 9, 10, 13, 22, 26}, {b, b}}, {{8}, {a, b, b, b}},
 {{6, 10, 13}, {b, b, a, b}}, {{17, 18}, {a, a, a, a}}}
```