Slide 19

Si L es recursivo, entonces es recursivamente numerable

```
program GeneraRec (output);
    procedure sgte (var x: string);
    (* calcula la siguiente x en el orden natural *)
    begin ... end;
var x: string;
Begin
     x := '';
     while (1=1) do begin
         if enL(x) then writeln (x);
         sgte (x)
     end
End.
```

Genera L en el orden natural[a] (de longitud creciente).

[a] ver ejercicios "para empezar"

Slide 20

Si L es recursivamente numerable, entonces existe una M.T, M tal que $LR(M) = L$ (aunque es posible que no siempre se pare)

```
program ReconoceRN (input, output);
var x , y : string;
   ...
Begin
    readln (x);
    y := x1 (* primera de L *) ;
    while (y <> x) do
      begin
          writeln ('Aún no lo sé');
          siguienteEnL (y)
      end
    writeln ('¡Si!')
End.
```

Slide 21

Dada una M.T, M, $LR(M)$ es recursivamente numerable.

```
procedure siguientePar (var i, j: integer);
(* genera sucesivamente (1,1),
                        (1,2),(2,1),
                        (1,3),(2,2),(3,1),
                        (1,4),(2,3),(3,2),(4,1)
                         ...*)
var s: integer;
begin    s:= i+j;
         if j=1 then begin i := 1;   j := s end
                else begin i := i+1; j := j-1 end
end;
```

Slide 22

```
program GeneraLRM (output);
   procedure siguientePar (var i, j: integer);
   (* genera sucesivamente pares de N x N  *)
var i, j : integer;
Begin
  i:= 1; j:= 1; x := '';
  repeat
     for k := 1 to i-1 do sgte(x);
     haz j pasos de M ;
     if (M ha dicho 'Si') then writeln (x);
     siguientePar (i,j)
  until (1=0)
End.
```

Genera $LR(M)$, no necesariamente en orden natural.

Slide 23

El complementario de un lenguaje recursivo es recursivo

```
function enComplL (x:string):boolean}
(* enComplL(x) = TRUE sii x no está en L *)
begin
    enComplL := not enL(x)
end;
```

La unión de lenguajes recursivos es recursiva

```
function enL1UL2 (x: string):boolean;
begin
   if      enL1(x) then enL1UL2 := TRUE
   else if enL2(x) then enL1UL2 := TRUE
   else                 enL1UL2 := FALSE
end;
```

Slide 24

La unión de lenguajes rec. numerables es rec. numerable.

```
program ReconoceL1UL2 (input, output);
var i = 1 : integer;
Begin
   readln (x);
   repeat
       i pasos de RecRNL1(x); i pasos de RecRNL2(x);
       if RecRNL1 ha parado en 'Si' then
                   writeln ('Si'); goto FIN;
       if RecRNL2 ha parado en 'Si' then
                   writeln ('Si'); goto FIN;
   until (1=0);
Fin: End.
```

Si un lenguaje y su complementario son rec. numerables, ambos son recursivos:

Slide 25

```
function enL (x : string): boolean;
var i =1 : integer;
begin repeat
       i pasos de MR(L); i pasos de MR(Comp(L));
       if MR(L) ha parado en 'Si' then
                    enL:= TRUE; goto FIN;
       if MR(L) ha parado en 'No' then
                    enL:= FALSE; goto FIN;
       if MR(Comp(L)) ha parado en 'Si' then
                    enL:= FALSE; goto FIN;
       if MR(L) ha parado en 'No' then
                    enL:= TRUE; goto FIN;
       i := i+1;
   until (1=0);
FIN: end;
```

Slide 26

Dado un lenguaje L, una y sólo una de las siguientes afirmaciones es cierta:

1. L y $\overline{L}$ son recursivos
2. ni L ni $\overline{L}$ son recursivamente numerables
3. L es recursivamente numerable y no recursivo y $\overline{L}$ no es recursivamente numerable
4. $\overline{L}$ es recursivamente numerable y no recursivo y L no es recursivamente numerable

Slide 27

Codificación de Máquinas de Turing

$Q = \{q_1, q_2, \ldots, q_n\}$ $\Gamma = \{0, 1, \hbar\}$ $\Sigma_E = \{0, 1\}$ $F = \{q_2\}$

$X_1 := 0$ $X_2 := 1$ $X_3 := \hbar$ $D_1 :=\leftarrow$ $D_2 :=\rightarrow$

$f(q_i, X_j) = (q_k, X_l, D_m)$ se codifica $1^i01^j01^k01^l01^m$

Una M.T. completa M (su tabla de transición) se codifica:

$< M >=$
$1^{i_1}01^{j_1}01^{k_1}01^{l_1}01^{m_1}001^{i_2}01^{j_2}01^{k_2}01^{l_2}01^{m_2}00\ldots1^{i_p}01^{j_p}01^{k_p}01^{l_p}01^{m_p}$

Una cadena de Σ_E, análogamente
$< x >=< X_{i_1}X_{i_2}\ldots X_{i_q} >= 1^{i_1}01^{i_2}0\ldots01^{i_q}$

El conjunto de máquinas de Turing es numerable
El conjunto de lenguajes, no
Existen lenguajes que no son recursivamente numerables

Slide 28

Máquina de Turing Universal

Tres cintas:

- una para $< M >$ (programa almacenado)
- otra para la cinta de M (codificada)
- otra para el estado. Inicialmente $< q_1 >= 1$

Entrada: $< M, w >$, que copia respectivamente en las cintas 1 y 2

Algoritmo: MTU ejecuta sobre la segunda cinta las instrucciones almacenadas en la primera.

MTU es un computador de propósito general:

$MTU(< M, w >)$ se parará sii $M(w)$ se para.
Lo hará con 11 en la tercera cinta sii M lo habría hecho en q_2
Dejará en la segunda cinta $< M(w) >$

Slide 29

$\mathcal{MT} = \{MT_1, MT_2, \ldots, MT_j, \ldots\}$

$\Sigma_E^* = \{w_1, w_2, w_3, \ldots, w_i, \ldots\}$

T	M_1	M_2	$\ldots$	M_j	$\ldots$
w_1	0	0	$\ldots$	1	$\ldots$
w_2	0	1	$\ldots$	0	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
w_i	1	1	$\ldots$	1	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$

$T[i,j] = 1$ sii $M_j(w_i) \perp$ aceptando

$L_U = \{w_i \ / \ T[i,i] = 0\}$

L_U no es recursivamente numerable

Slide 30

$L_D = \overline{L_U} = \{w_i \ / \ T[i,i] = 1\}$

L_D no es recursivo

L_D es recursivamente numerable:

```
program ReconoceL_D (input, output);
var x : string; n: integer;
Begin
    readln (x);
    n := número de orden de x (* x=w_n *) ;
    construir la MT número n (* MT_n *)
    ejecutar MTU (MT_n,w_n)
    if ha aceptado then
          writeln ('Si')
End.
```

Slide 31

Si L es de tipo 0, es recursivamente numerable

Si L es recursivamente numerable, es de tipo 0

$$q_1 x \vdash^* \alpha q_f \beta$$
$$\alpha q_f \beta \Rightarrow^* q_1 x$$

$\Sigma_A := \Gamma \cup Q \cup \{A_\wedge, A_Q, S\}$

$$\begin{cases} S & \to & \hbar S \mid S\hbar \mid A_\wedge A_Q & \\ A_Q & \to & AA_Q \mid A_Q A \mid q & \forall A \in \Gamma - \{\hbar\}, q \in F \\ Bq_j & \to & q_i A & \text{si } f(q_i, A) = (q_j, B, \to) \\ q_j CB & \to & Cq_i A & \forall C \in \Gamma \text{ si } f(q_i, A) = (q_j, B, \leftarrow) \\ A_\wedge q_1 & \to & \varepsilon & \\ \hbar & \to & \varepsilon & \end{cases}$$

Slide 32

Si L es de tipo 1, es recursivo

Existe un lenguaje de recursivo que no es de tipo 1

T_G	G_1	G_2	$\ldots$	G_j	$\ldots$
w_1	0	0	$\ldots$	1	$\ldots$
w_2	0	1	$\ldots$	0	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
w_i	1	1	$\ldots$	1	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$

$$T_G[i,j] = 1 \qquad \text{sii} \qquad w_i \in L(G_i)$$

$$L_N = \{w_i \ / \ Ti_G[i,i] = 0\}$$

Autómatas linealmente acotados $\leftrightarrow$ Lenguajes tipo 1