# Clase #13 de 29
# Implementaciones de Mealy & Moore

*Jul 1, Martes*
*Jul 2, Miércoles*

# Máquinas de Mealy & Máquinas de Moore

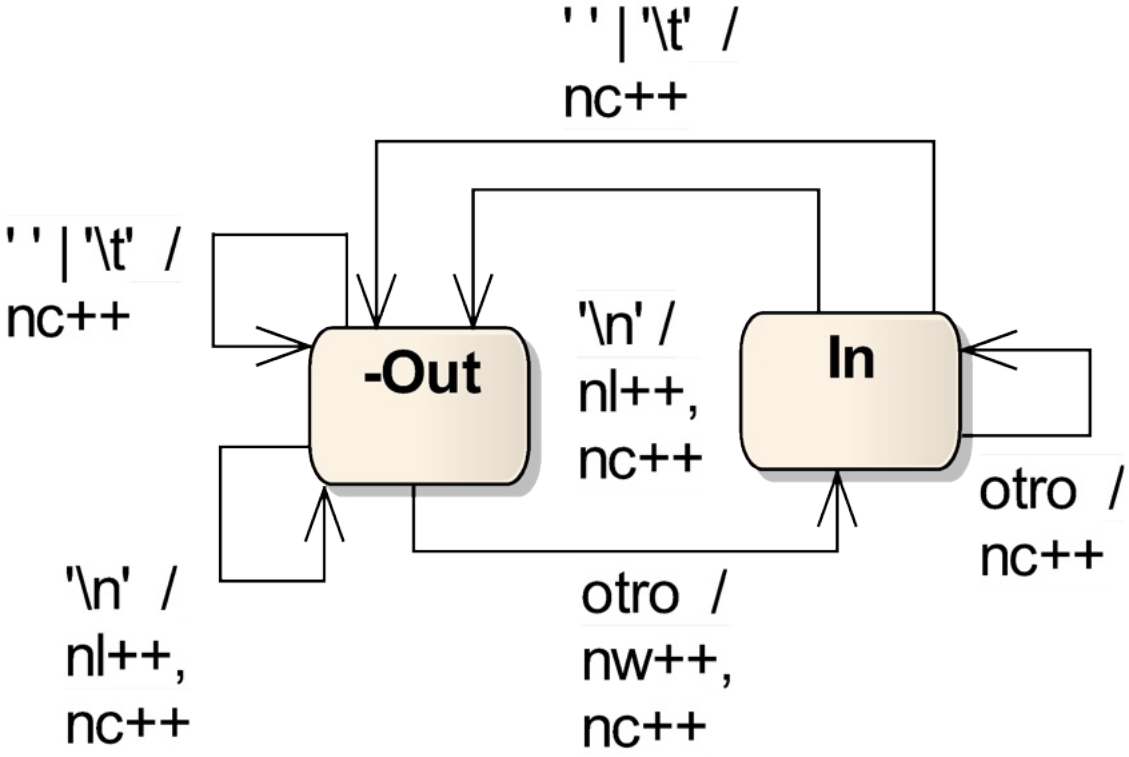
Máquinas de Estado Parte III

# Máquinas de Estado para especificar comportamiento

- ¿Cuál es el comportamiento?
- ¿Cómo describirlo?
  - ¿Prosa?
  - ¿Caso de uso?
  - ¿Función?
- Dominio e Imagen
  - $F : \Sigma_i^* \rightarrow \Sigma_o^*$
- ¿Como representar la función?
  - Diagrama de Transiciones.

# Formalización Matemática para Mealy

- ¿Es suficiente un Digrafo?
- ¿Es suficiente una Función?
  - $T : Q \times \Sigma i \rightarrow Q \times \Sigma o$
- Par ordenando (2-upla)
  - $(T : Q \times \Sigma i \rightarrow Q \times \Sigma o, \text{inicial})$
- 5-upla
  - Elementos
    - Estados
    - Estado inicial
    - Entradas
    - Salidas
    - Función
  - Definición
    - $M = (Q, eo, \Sigma i, \Sigma o, T)$
    - $T : Q \times \Sigma i \rightarrow Q \times \Sigma o$
- 6-upla
  - Función Acción separada
  - $M = (Q, eo, \Sigma i, \Sigma o, T, A)$
  - $T : Q \times \Sigma i \rightarrow Q$
  - $A : Q \times \Sigma i \rightarrow \Sigma o.$

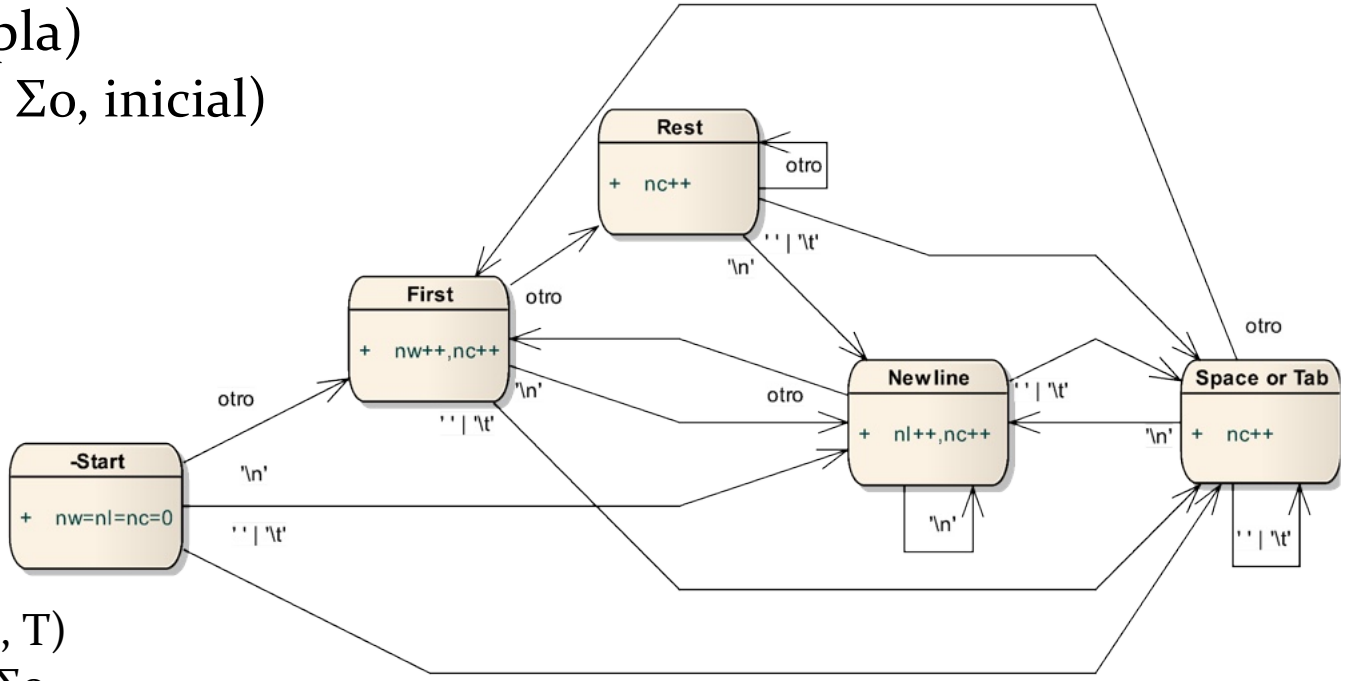


# Diferencias Moore y Mealy

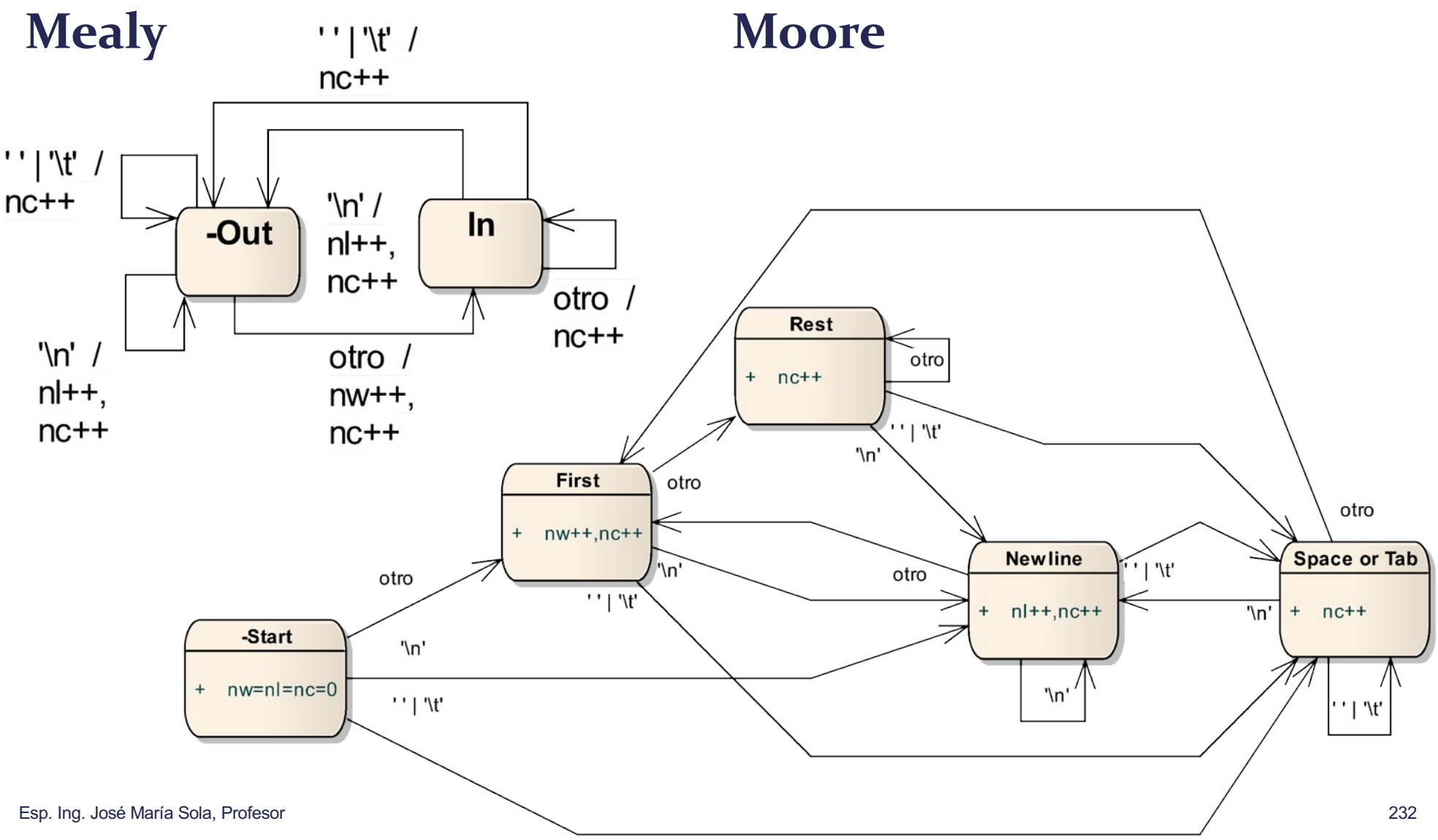
- Funciones
  - Mealy
    - $A : Q \times \Sigma i \rightarrow \Sigma o$
  - Moore
    - $A : Q \rightarrow \Sigma o$
  - General
    - $T : Q \times \Sigma i \rightarrow Q \times \Sigma o$
- Algoritmos particulares
  - Mealy
    - Seleccionar Estado
    - Seleccionar Carácter
    - Accionar
    - Actualizar Estado
  - Moore
    - Seleccionar Estado
    - Accionar
    - Seleccionar Carácter
    - Actualizar Estado
- Algoritmos generales
  - General
    - Seleccionar Estado
    - Accionar
    - Seleccionar Carácter
    - Accionar
    - Actualizar Estado
  - Un algoritmo un poco más abstracto
    - while there's another character
    - DoMoore(s)
    - DoMealy(s, c)
    - s = GetNextState(s, c)

# Formalización Matemática para Moore

- Par ordenando (2-upla)
  - (T : Q × Σi → Q × Σo, inicial)
- 5-upla
  - Elementos
    - Estados
    - Estado inicial
    - Entradas
    - Salidas
    - Función
  - Definición
    - M = (Q, eo, Σi, Σo, T)
    - T : Q × Σi → Q × Σo
- 6-upla
  - Función Acción separada
  - M = (Q, eo, Σi, Σo, T, A)
  - T : Q × Σi → Q
  - A : Q → Σo.



# Moore y Mealy – Modelos Equivalentes



# Implementaciones de Máquinas de Estado

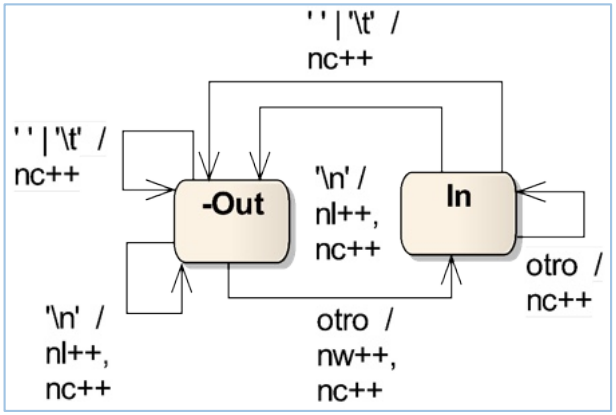
# Implementación #1 – Estados como variable entera y Transiciones como selección estructurada y actualización de variable

- Algoritmo de la implementación #1 para "máquina de Mealy"
  - Mientras haya caracteres
  - Seleccionar Estado
  - Seleccionar Carácter
  - Accionar
  - Actualizar Estado
- ¿Cuál es la abstracción aplicada?
  - Abstracción de datos
  - Abstracción del control de flujo
- Implementación #1
  - Estado como **variable entera**
  - Transiciones como **selección estructurada y actualización de variable**

- ¿Con qué estructura de control de flujo la implementaría? ¿Con if o con switch?¿Por qué?
- Eficiencia de If y de Swtich
  - Secuencia de comparaciones
  - Tabla de salto
- Implementación #1 con if
- Implementación #1 con switch

# Implementación #1 – If versus Switch

```
while ((c = getchar()) != EOF)
   if (state == OUT )
      if (c == '\n'){
         ++nl, ++nc;
         state = OUT;
      }
      else if (c==' ' || c=='\t'){
         ++nc;
         state = OUT;
      }
      else {
         ++nw, ++nc;
         state = IN;
      }
   else /* IN */
      if (c == '\n'){
         ++nl, ++nc;
         state = OUT;
      }
      else if (c==' ' || c=='\t'){
         ++nc;
         state = OUT;
      }
      else {
         ++nc;
         state = IN;
      }
printf("%d %d %d\n", nl, nw, nc);
return 0;
```




```
while ((c = getchar()) != EOF)
   switch(state){
      case OUT:
         switch(c){
            case '\n':
               ++nc, ++nl;
               state = OUT;
               break;
            case ' ' :
            case '\t':
               ++nc;
               state = OUT;
               break;
            default:
               ++nc;
               state = IN;
         }
         break;
      default:/* IN */
         switch(c){
            case '\n':
               ++nc, ++nl;
               state = OUT;
               break;
            case ' ' :
            case '\t':
               ++nc;
               state = OUT;
               break;
            default:
               ++nc;
               state = IN;
         }
   }
printf("%d %d %d\n", nl, nw, nc);
return 0;
```

# Más Implementaciones

- Implementación #1
  - Estados como **variable enum**
  - Transiciones como **selección estructurada y actualización de variable**
- Implementación #2
  - Estados como **etiquetas**
  - Transiciones como **selección estructurada y saltos incondicionales**
- Implementación #3
  - Estados como **funciones recursivas**
  - Transiciones como **selección estructurada e invocaciones**
- Implementación #4
  - Estado como **variable entera**
  - Transiciones como **tabla**
    - Matriz de pares
      - Arreglo de arreglo de estructuras de dos campos
- Implementación #5
  - Estados como **funciones y variable puntero a función actual**
  - Transiciones como **selección estructurada, actualización de variable e invocaciones**
- Implementación #6
  - Estados como **variable enum**
  - Transiciones como **función GetNextState(s,c)** y **DoMoore(s)** ó **DoMealy(s,c)**
- Otras…

# Implementación #2 – Estados como etiquetas y transiciones como saltos

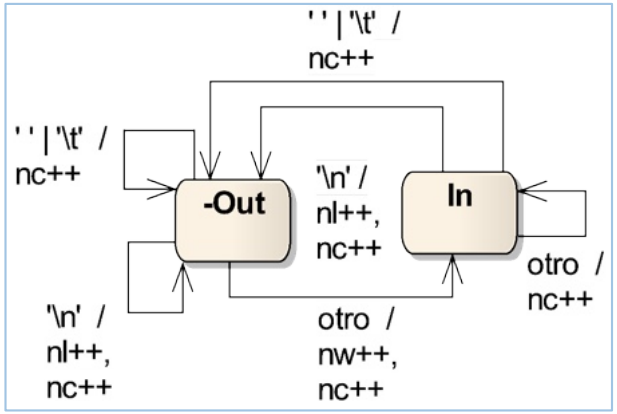
```
int main(void){
  int nl, nw, nc;

  nl = nw = nc = 0;
  goto Out;

  Out:
    switch( getchar() ){
      case '\n':
        ++nl, ++nc;
        goto Out;
      case ' ' :
      case '\t':
        ++nc;
        goto Out;
      case EOF:
        goto End;
      default:
        ++nw, ++nc;
        goto In;
    }

  In:
    switch( getchar() ){
      case '\n':
        ++nl, ++nc;
        goto Out;
      case ' ' :
      case '\t':
        ++nc;
        goto Out;
      case EOF:
        goto End;
      default:
        ++nc;
        goto In;
    }

  End:
    printf("%d %d %d\n", nl, nw, nc);
    return 0;
}
```



# Ejercicios de K&R que se resuelven con Máquinas de Estado

- 1-12. Escriba un programa que imprima su entrada de a una palabra por línea
- 1-9. Escriba un programa que copie su entrada en su salida, remplazando cada secuencia de uno o más blancos por un solo blanco
  - Diseñar con máquina de Mealy
  - Diseñar con máquina de Moore
- 1-23. Escriba un programa que remueva todos los comentarios de un programa C. No se olvide de tratar correctamente las cadenas y los caracteres literales. Los comentarios en C no se anidan.

# ¿Consultas?

# Fin de la clase