



Informática

Curso 2024

Facultad de Ingeniería

Universidad de la República

Instrucciones

- **Asignación**
 - Asignan valores a variables.
- **Estructuras de Control**
 - Organizan el curso (flujo) de ejecución.
 - Pueden basarse en Condiciones

Asignación

- El objetivo de una sentencia de asignación es cambiar el valor almacenado en una variable.

`<variable> = <expresión>`

- Ejemplos:

`x = 3 + 4`

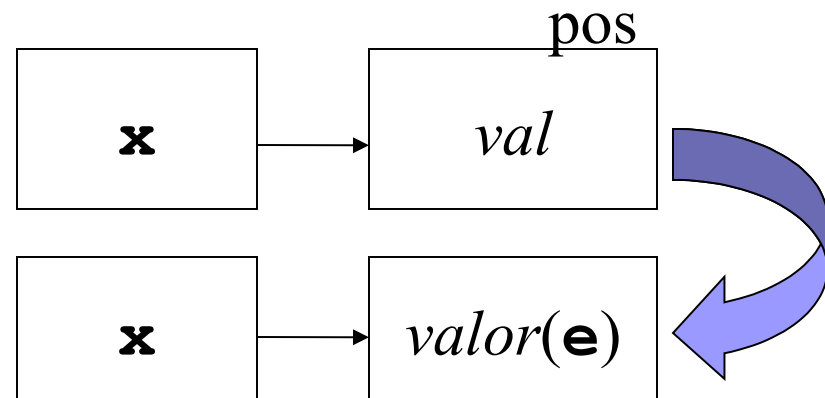
`x = x + (3.0 / 5.9)`

`c = 'a'`

Asignación

- Una asignación $x = e$ es ejecutada siguiendo estos pasos:

- 1.- Se evalúa la expresión e
- 2.- Se reemplaza el valor almacenado en la posición de memoria pos , correspondiente a la variable x , por el valor de e .



Asignación

- La ocurrencia de una variable en el lado izquierdo de una asignación denota la posición de memoria donde almacenar el valor resultante de evaluar la expresión en el lado derecho.

$$**x = 1;**$$

- La ocurrencia de una variable en el lado derecho de una asignación denota su valor actual.

$$**y = x + 1;**$$

Asignación

- Una misma variable puede aparecer en la parte izquierda y derecha de una asignación.

$$**x = x + 1**$$

- Esto NO debe interpretarse como una ecuación matemática!
- Sólo significa que estamos usando el valor actual de la variable x para calcular su nuevo valor.

Estructuras de control

- Mecanismos para definir el orden en que se ejecutan las instrucciones de un programa.
- Legibilidad y Mantenimiento de los programas.

Estructuras de control

■ Nivel de instrucción

- Especifican flujo de control entre instrucciones de programa.
 - Secuencia
 - Selección
 - Iteración

■ Nivel de Unidad

- Especifican flujo de control entre unidades de programa.
 - Subprogramas

Estructuras de control

■ Nivel de instrucción

- Especifican flujo de control entre instrucciones de programa.

- Secuencia 

- Selección

- Iteración

■ Nivel de Unidad

- Especifican flujo de control entre unidades de programa.

- Subprogramas

Estructuras de control

Secuencia

- La ejecución de las instrucciones se hace en un orden dado, en forma secuencial
 - Una después de la otra
 - No se ejecuta la segunda hasta que la primera haya terminado de ejecutarse, y así sucesivamente.
- Se sigue el orden normal de lectura de izquierda a derecha y de arriba a abajo.

Estructuras de control

Secuencia

■ Sintaxis

<<instrucción1>>;

<<instrucción2>>

■ Semántica:

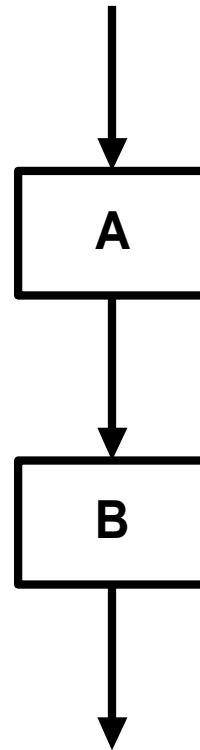
- Indica que *<<instrucción2>>* se ejecutará a continuación de *<<instrucción1>>*

Estructuras de control

Secuencia

■ Diagrama de flujo

- A - Instrucciones
- B - Instrucciones



Estructuras de control


Secuencia

$x = 1;$

$x = x + 1;$

Estructuras de control

■ Nivel de instrucción

- Especifican flujo de control entre instrucciones de programa.
 - Secuencia
 - Selección 
 - Iteración

■ Nivel de Unidad

- Especifican flujo de control entre unidades de programa.
 - Subprogramas

Estructuras de control

Selección

- Especificar selección entre distintas posibilidades.
- Emplea expresiones lógicas para decidir si una instrucción o conjunto de instrucciones se ejecutará o no.
- Instrucciones
 - *if*
 - Dos alternativas
 - *elseif*
 - Más de dos alternativas

Estructuras de control

Selección – Dos alternativas

- Instrucción *if*

- Semántica

- Selección entre dos alternativas, según la evaluación de una condición lógica.

Estructuras de control

Selección – Dos alternativas

■ Sintáxis

```
if <<condición>>  
    <<instrucción1>>  
else  
    <<instrucción2>>  
end
```

■ Lógica

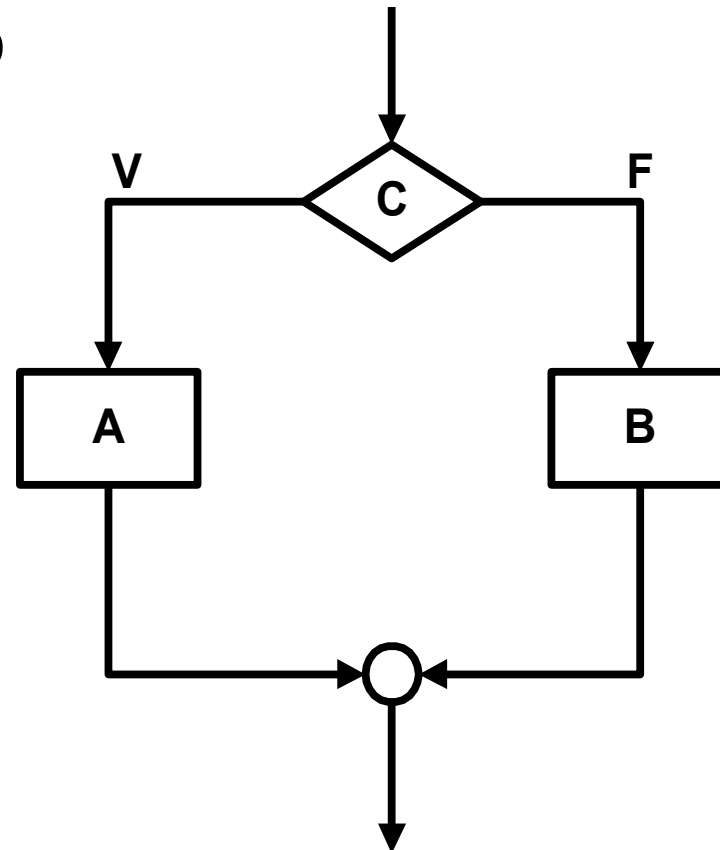
- Se evalúa la condición
- Si es verdadera se ejecuta instrucción1
- Sino se ejecuta instrucción2

Estructuras de control

Selección – Dos alternativas

■ Diagrama de Flujo

- C - Condición
- A - Instrucciones
- B - Instrucciones



Estructuras de control

Selección – Dos alternativas

```
if x  $\neq$  0  
    y = y/x;  
else  
    y = y/4;  
end
```

Estructuras de control

Selección – Dos alternativas

- Cuanto valen x e y al final de la ejecución

```
x = 1;  
y = 2;  
if x > y  
    x = x + 1;  
else  
    y = y + 1;  
end
```

Estructuras de control

Selección – Dos alternativas

- El ***else*** es opcional.

```
if <<condición>>  
    <<instrucción1>>  
end
```

Estructuras de control

Selección – Dos alternativas

- Pueden anidarse

```
if <<condición1>>  
    if <<condición2>>  
        <<instrucción1>>  
    else  
        <<instrucción2>>  
    end  
else  
    <<instrucción3>>  
end
```

Estructuras de control

Selección – Dos alternativas

■ Anidada

```
if a > b  
    if a > c  
        resultado = a;% A es el grande  
    end  
end
```

■ Sin anidar

```
if (a > b) & (a > c)  
    resultado = a;% A es el grande  
end
```

Estructuras de control

Selección – Más de dos alternativas

■ Anidamiento de *if*

```
if (calif == 'D') | (calif == 'F')
    nota = 0;% Trabajo deficiente
else
    if (calif == 'C') | (calif == 'B')
        nota = 6;% Buen trabajo
    else
        if calif == 'A'
            nota = 12;% Excelente trabajo
        end
    end
end
end
```


Estructuras de control

Selección – Más de dos alternativas

■ Instrucción ***elseif***

```
if <<condicion1>>
```

```
  <<instrucción1>>
```

```
elseif <<condicion2>>
```

```
  <<instrucción2>>
```

```
elseif <<condicion3>>
```

```
  <<instrucción3>>
```

```
end
```

Estructuras de control

Selección – Más de dos alternativas

```
if (calificacion >= 25) & (calificacion < 60)
    nota = 4; % Curso Aprobado
elseif (calificacion < 25)
    nota = 0; % Curso Reprobado
elseif (calificacion >= 60)
    nota = 6; % Exoneración
end
```

Pregunta interesante

Selección

- ¿Qué es mejor usar ifs en secuencia o if-elseif?

```
if (calificación >= 25)and(calificación < 60)
    nota = 4; % Curso Aprobado
endif
if (calificación < 25)
    nota = 0; % Curso Reprobado
endif
if (calificación >= 60)
    nota = 6; % Exoneración
endif
```

Pregunta interesante

Selección

- ¿Qué es mejor usar ifs en secuencia o if-elseif?

```
if (calificación >= 25)and(calificación < 60)
    nota = 4;% Curso Aprobado
elseif (calificación < 25)
    nota = 0;% Curso Reprobado
elseif (calificación >= 60)
    nota = 6; % Exoneración
endif
```

Pregunta interesante

Selección

- ¿Qué es mejor usar ifs en secuencia o if-elseif?
- El if-elseif es más expresivo que los ifs en secuencia.
- Usando if-elseif el código es más claro ya que se aprecia fácilmente que los casos son excluyentes, mientras que usando ifs hay que revisar caso a caso.