

Principios de Lenguajes de Programación

Control de Secuencias: Estructuras de Control a Nivel Instrucción

Facultad de Informática
Universidad Nacional del Comahue

Primer Cuatrimestre



Índice

- Introducción
- Sentencias de control
 - Definición y particularidades
- Implementación
- Ejemplos

Bibliografía

- Sebesta cap 8 Pratt cap 6

Introducción

- Niveles de las Estructuras de Control (paradigma imperativo):
 - **Dentro de las Sentencias/Instrucciones** (ie. reglas de precedencia y paréntesis).
 - ***Entre sentencias (instrucciones) o grupos de sentencias***, (tales como condicionales o iteraciones)
 - **Entre subprogramas** (como abstracciones), tales como llamadas a programas, corutinas, etc.



Estructuras de control

- Llamamos ***estructura de control*** a una instrucción de control y a las instrucciones que se ejecutan como parte de ese control
- Por definición cualquier algoritmo puede describirse utilizando tres estructuras de control básicas:
 - **Secuencias** (Composición)
 - **Selección Simple** (alternativa de dos vías)
 - **Iteración** con prueba lógica inicial (pretest)

Asignación

- La sintaxis general tiene la siguiente forma:
`<variable destino> <operador de asignación> <expresión>`
- El operador de asignación varia de lenguaje de programación en lenguaje de programación.

`A:= B` Algol, Pascal, Ada

`A = B` C, Fortran, Java

`MOVE B TO A` Cobol

`A ← B` APL

Asignación

Se puede definir la operación de asignación como:

- 1) Computar el valor del lado izquierdo (l-value)
- 2) Computar el valor de la expresión del lado derecho (r-value)
- 3) Asignar el valor computado del lado derecho al computado como objeto de dato del lado derecho
- 4) Retornar el valor computado como valor del lado derecho como resultado de la asignación

l-value: la locación de memoria de un objeto de dato

r-value: el valor del objeto de dato

La semántica de la operación de asignación puede pensarse como:

- Copia de la referencia
- Dereferenciamiento implícito

Asignación

La operación de asignación puede pensarse como:

- una instrucción (no retorna ningún valor)
- una expresión

Instrucción	Pascal, Ada, Algol, Fortran, PL/1
Expresión	C, Java, APL, ML, Snobol4

- Así por ejemplo, en C uno puede escribir la siguiente asignación:

$C = (B = 2)$

- La cual se interpreta como se le asigna el valor de 2 a B. El resultado de la expresión $B=2$, es decir 2, se la asigna a C. Por lo cual se le asigna 2 a C

Estructuras de control: Selección

- Selección de dos vías

`If (expresión de control)`

`Then Cláusula Entonces (verdadera)`

`Else Cláusula Sino (falso)`

- Consideraciones:
 - Tipo y formato de la “expresión de control”
 - Booleano / entero, parentesis
 - Cómo se especifican las Cláusulas (then/else)
 - Simples y compuestas, uso de Begin/End o {}, indentación (Phyton)
 - Qué ocurre con las expresiones anidadas
 - Uso de Endif / convención de “más próximo”.

Estructuras de control: Selección

- Python

```
if x>y:
    x=y
    print "case1"
```

- Java

```
if (sum == 0)
    if (count == 0)
        result=0;
else
    result=1;
```

```
if (sum == 0) {
    if (count == 0)
        result=0;
}
else
    result=1;
```

- Rubi

```
if (sum == 0)
    if (count == 0)
        result=0;
else
    result=1;
end
end
```

```
if sum == 0:
    if count == 0:
        result=0;
else:
    result=1;
```

Estructuras de control:

Selección

- Selección de vías múltiples

```
caso (expresión)
    Valor_01: sentencia(s);
    ...
    Valor_n : sentencia(s);
    Default : sentencia(s);
fin
```

- Consideraciones:
 - Formato y Tipo de Expresión (de control)
 - Cómo se especifican el valor y las sentencias elegibles
 - Pueden elegirse sentencias de más de un valor elegible
 - Qué ocurre con los valores no expresados

Estructuras de control: Selección

```
switch (index) {  
    case 1:  
  
    case 3: odd+=1;  
           sumodd+=index;  
           break;  
  
    case 2:  
  
    case 4: even+=1;  
           sumeven+=index;  
           break;  
    default: printf("error");
```

```
switch (value) {  
    case -1:  
        negative++;  
        break;  
    case 0: Zeros++;  
           goto case 1;  
    case 1:  
        positives++;  
    default: printf("error");  
  
case expression is  
    when choice list => S1;  
    ...  
    when choice list=> Sn:  
        [when others => So;]  
end case;
```



Estructuras de control: Repetición

- Repeticiones por Iteraciones o por Recursividad
- Distintos Tipos de Control para ***iterar***:
 - Control basado en ***contador***
 - Control basado en ***prueba lógica***
 - Prueba anterior
 - Prueba posterior
 - Mixto
 - Control basado en Estructura de Datos



Iteraciones: Control por Contador

- Sentencia con “*variable*” de iteración y forma de especificar valor *inicial*, *final* y *paso*
- Consideraciones
 - Tipo y alcance de la variable de iteración
 - Evaluación de Parámetros: una vez o en cada ciclo
 - Cambio del control por asignación de valores para parámetros de la Estructura (contador, expresión inicial, expresión final, etc.)

Iteraciones: Control Lógico

- El control de las iteraciones se basa en una expresión lógica (condición de salida)
- Consideraciones:
 - Posición prueba lógica: anterior/posterior
 - Si es una sentencia nueva o caso especial de control por contador
 - Control del programador (continuación/continue, salida dentro del bloque/break, etc.)

Iteraciones

- El número de elementos de una Estructura de Datos define el control de la iteración.
- Mecanismo de control como llamada a una función “iterador” que devuelve el próximo elemento en un orden determinado (si existe)
- Consideraciones:
 - Cómo se especifica la estructura de datos / tipos de datos del iterador
 - Cómo se especifica la función iterador: elemento actual, próximo elemento

Iteraciones

- Ejemplo Java:

```
public class Main {  
    public static void main(String[] args) {  
        int[] intary = { 1,2,3,4};  
        forDisplay(intary);  
        foreachDisplay(intary);  
    }  
    public static void forDisplay(int[] a){  
        System.out.println("Display an array using for  
loop");  
        for (int i = 0; i < a.length; i++) {  
            System.out.print(a[i] + " ");  
        }  
        System.out.println();  
    }  
    public static void foreachDisplay(int[] data){  
        System.out.println("Display an array using for  
each loop");  
        for (int a : data) {  
            System.out.print(a+ " ");  
        }  
    }  
}
```


Iteraciones: Control por Estructura de Datos


- Ejemplo C#:

```
class ForEachTest {  
    static void Main(string[] args)  
    {  
        int[] fibarray = new int[] {0, 1, 2, 3, 5, 8, 13};  
        foreach (int i in fibarray) {  
            System.Console.WriteLine(i);  
        }  
    }  
}
```

Iteraciones: Control por Estructura de Datos

- Ejemplo Python (listas, tuplas, cadenas)

```
>>> L = [1, 2, 3]
>>> for x in L:
    print(x ** 3, end=' ')
1 8 27 64 125
>>>
>>> for x in (1, 2, 3, 4, 5):
    print(x ** 3, end=' ')
1 8 27 64 125
>>>
>>> for x in 'Beethoven':
    print(x * 3, end=' ')
BBB eee eee ttt hhh ooo vvv eee nnn
>>>
```



Repetición: Salto Incondicional

- Transferencia a cualquier lugar del programa: (se especifica con una etiqueta o una posición relativa)
- Debate de los 70 y 80
- Problema principal: legibilidad
- Soporte:
 - Algunos no lo incluyen (java)
 - Otros lo restringen (C#) (gotos en switch, etc.)
 - Algunos lo “camuflan”: ie *loop*



Conclusion

- Amplia variedad de estructuras de control a nivel sentencia
- La elección de las propuestas de control (selección/iteración) depende del tamaño del lenguaje, legibilidad y escribibilidad.
- Los lenguajes de programación que adhieren a los paradigmas funcional y lógico tienen estructuras de control un poco diferentes
- La implementación depende de la complejidad de la estructura de control del lenguaje y de los recursos del lenguaje de la Máquina Virtual subyacente