領域與範圍

資科系
林偉川

基本概念

- 資料控制的意義
  - 程式執行到一個區塊時，對當時資料存取的管制及資料流向的控制即稱為資料控制
- 參用環境(referencing environment)
  - 當程式呼叫副程式時會產生一個參用環境

基本概念

- 參用環境建立的過程
  - 將實際參數(actual parameter)的資訊傳給型式參數(formal parameter)
  - 副程式建立區域變數
  - 當程式流程轉移給副程式時，首先會先參用自己的區域環境，若區域環境處找不到變數的定義，再到原呼叫程式處引用非區域變數

3

基本概念

- 假如副程式又呼叫了另一個副程式，此時將產生另一個參用環境
- 由副程式返回原呼叫程式時，參用環境將回復成原來狀況
- 此時原呼叫程式執行時無法引用副程式內的區域變數

4

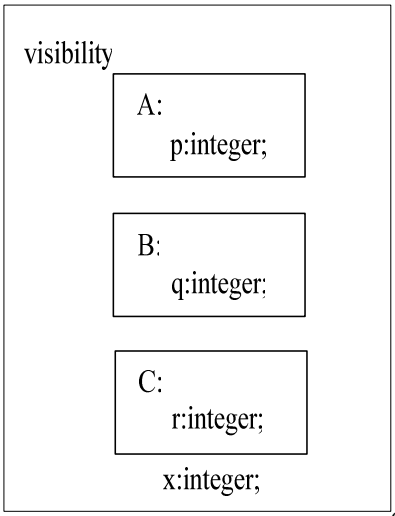
可見性(visibility)

- 某段程式可參用的變數對該程式而言即為可見，反之即為不可見
-
```
program visibility;
var x : integer;
  procedure A (p : integer);
    begin
     ...
    end;
  procedure B (q : integer);
    begin
     ...
    end;
  procedure C (r : integer);
    begin
     ...
    end;
begin
...
end.
```

5

可見性(visibility)

- 根據左側程式可得右邊的區塊圖：
- 對主程式而言x是可見；p，q，r非可見
- 對副程式A而言x，p為可見，而q，r非可見
- 對副程式B而言x，q為可見，而p，r非可見
- 對副程式C而言x，r為可見，而p，q非可見



6

領域與範圍

- 領域(scope) (空間)
  - 代表一段程式本文
- 範圍(extend) (生命週期)
  - 代表在某段時間內，變數會與儲存其值的儲存體做繫結的動作
- 對程式設計師來說，領域其實就是指一段空間
- 一個變數的領域其實是指這個變數在這段程式內是可被引用的(reference)

7

靜態領域法與動態領域法

- 靜態領域法
  - 在區塊中未定義的變數，根據程式的結構，在程式中包含其區塊的上一層區塊處尋找此變數是否在此區塊內定義
    - 若找到變數的宣告則變數即是在此區塊內定義
    - 若未找到則繼續往外層尋找直到變數第一次宣告處為止
    - 若搜尋完整個程式仍未找到變數之宣告，則該變數即為未定義之變數

8

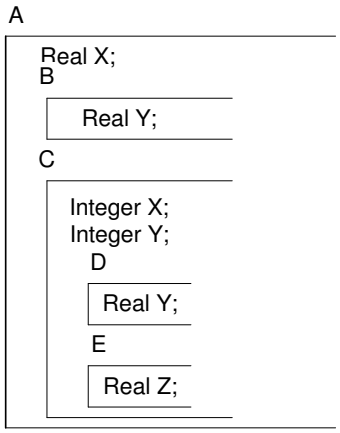
靜態領域法與動態領域法

- 動態領域法
  - 又稱為流動繫結法(fluid binding)
  - 變數的型態會在目前的區塊中尋找；尋找的順序為副程式被呼叫的反順序

9

實際範例一

- 就右側採用靜態領域法的程式中，指出每個區塊中所有界定的變數及其資料型態
- Block A: Real X
- Block B: Real X,Y
- Block C: Integer X,Y
- Block D: Integer X, Real Y
- Block E: Integer X,Y, Real Z



10

實際範例二

- 請列出列號6，9，13，16，程式執行時，所能使用的變數名稱及其資料型態，

  以「變數名稱：資料型態」表示
- 6:Real X,Y,W, Integer Z
- 9:Real X,Y, Integer V,W,Z
- 13: Integer X,Y,Z
- 16:Char X,Z, Integer Y

```
1.PROCEDURE A;
2.VAR X, Y, Z : INTEGER;
3.
4.  PROCEDURE B;
5.  VAR W, X, Y : REAL;
6.
7.    PROCEDURE C;
8.    VAR V, W : INTEGER;
9.
10.   END {C}
11.
12.  END {B}
13.
14.  PROCEDURE D;
15.  VAR X, Z : CHAR;
16.
17.  END {D}
18.
19.END {A}
```

11

實際範例三

- 右側的程式

(1) 採取靜態領域，副程式A所列印的x之值為何？5

(2)採取動態領域，副程式A所列印的x之值為何？10

```
program MAIN；
var x:integer;
procedure A;
begin
  writeln('x=',x)
end; {of procedure A}
procedure B;
Var x: integer;
begin
  x:=10;
  A
end; {of procedure B}
begin {of main}
  x:=5;
  B
end.
```

12

領域的間隙

- 定義
  - 採用靜態領域法的情況下，不正常的否定先前變數的宣告即稱為領域的間隙(hole in scope)

13

領域的間隙範例

- program hole_in_scope;
  var x:real;
  procedure A;
  begin
  x:=x × x ;
  write(x)
  end;
  procedure B ;
  var x:integer;
  begin
  A
  end;
  begin
  B
  end.

- 在副程式B中，變數x宣告為整數，而在B中呼叫了副程式A，但在主程式中變數x宣告為實數，由靜態領域法可知此時x須使用的型態為實數。此種不正常的否定先前變數宣告的現象即稱之為領域的間隙

14

區域性資料

- 某個區域內所能引用的資料即稱為區域性資料
- 區域性資料僅適用某個區域內
- 區域變數處理的方法有保留法及清除法

15

區域性資料處理方法

- 保留法(retention)
  - 當副程式結束其執行動作時，會將副程式內區域變數的值保留；當副程式再次被呼叫時會沿用該保留的值做為程式執行之依據
  - 如，Fortran、C、C++、Java及Cobol
  - 採用靜態儲存區配置法(static storage allocation)，也就是在編譯時將記憶體的空間配置給區域變數使用

16

區域性資料處理方法

- 清除法(deletion)
  - 當副程式結束其執行動作時，會將副程式內區域變數的值清除，因此每次呼叫該副程式時其區域變數的初值是固定的
  - 如，Pascal、Ada、Lisp、APL、C、C++、Java與SNOBOL
  - 採用動態儲存區配法(dynamic storage allocation)，也就是在執行時將記憶體的空間配置給區域變數使用

17

範例

- 試就右側之程式段分別以清除法及保留法各別說明執行結果為何？
- 保留法:5, 10, 10, 15
- 清除法:5, 10, 5, 10

```
procedure B;
var P:integer=5;
 begin
    write(P);
    P:=P+5;
    write(P);
 end;{B}
```

```
procedure A;
begin
     ....
    B;
     ....
    B;
end;{A}
```

18

活動記錄

- 活動記錄(activation record)主要是用在副程式呼叫時
- 每次作副程式呼叫時就會產生一個相對應的活動記錄，其內記錄了副程式在執行的過程中所有可能參用的資訊

19

活動記錄

- 活動記錄的內容
  - 靜態鏈、動態鏈、返回位址、型式參數、區域變數、算術運算暫存值、參數傳遞暫存值、函數傳回值及符號資料
  - 符號資料主要是記錄此區段所宣告的副程式的名稱及標記(label)
  - 靜態鏈是指活動記錄的一個欄位，副程式的靜態鏈會指向包含其區塊的活動記錄
  - 動態鏈也是活動記錄的一個欄位，副程式的動態鏈會指向呼叫它的副程式的活動記錄

20

活動記錄範例

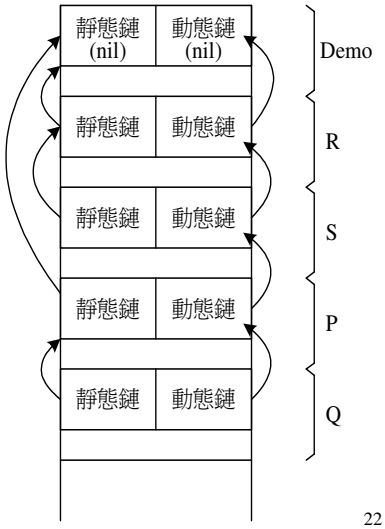
- program Demo ;
  procedure P;
  begin
  ...
  procedure Q;
  begin
  ...
  end;{of Q}
  ...
  Q;
  end;{of P}

- procedure R;
  begin
  ...
  procedure S;
  begin
  ... P;
  end;{of S}
  ... S;
  end;{of R}
  begin
  R;
  end.{of demo}

活動記錄範例

- 執行時程式段被呼叫的先後順序為：

Demo→R → S → P → Q



顯示堆疊

- 顯示堆疊是在採用靜態領域法時，尋找變數在何處定義的方法
- 顯示的內容是由許多指標組成，這些指標指向目前區塊的活動記錄及包含目前區段之所有區段的活動記錄

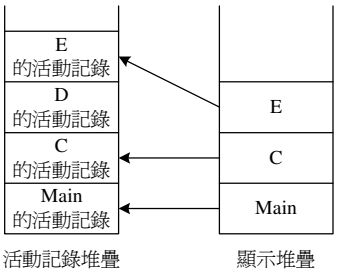
23

顯示堆疊範例

- procedure main;
  procedure A;
  procedure B;
  ….
  end B;
  end A;

  procedure C;
  procedure D;
  …
  end D;
  procedure E;
  …
  end E;
  end C;
  end main;

- Main→ C → D → E



24

動態領域法的實作

- 動態領域法的實作方式有二種
  - 深存取(deep access) ：用動態鏈沿著呼叫反順序來尋求變數之定義
  - 淺存取(shallow access) ：用中央環境表(變數名稱、變數值、旗標1表示可見)及隱藏堆疊(存放目前區段中定義之變數同名且可見的變數名稱與值)表示執行時非本地環境。

25

動態領域法範例：深存取

- main begin real x,y,z;
  procedure P;
  real I,j,k
  procedure Q;
  real a,b,x,s;
  R;
  end Q;
  end P;
  procedure R;
  real b,c,k,s,t;
  …
  end R;
  P;
  end main;

- Main→ P → Q → R

Main

X	Real
Y	Real
z	Real

Main→ P

X	Real
Y	Real
z	Real
I	Real
J	Real
K	Real

26

動態領域法範例

Main→ P → Q

X	Real
Y	Real
z	Real
I	Real
J	Real
K	Real
A	Real
B	Real
X	Real
s	Real

Main→ P → Q → R

X	Real
Y	Real
z	Real
I	Real
J	Real
K	Real
A	Real
B	Real
X	Real
s	Real

b	Real
c	Real
k	Real
s	Real
t	Real

27

動態領域法範例：淺存取中央環境表

X	1	main	X	1	main	X	1	Q	X	1	Q
Y	1	main	Y	1	main	Y	1	main	Y	1	main
Z	1	main	Z	1	main	Z	1	main	Z	1	main
I	0		I	1	P	I	1	P	I	1	P
J	0		J	1	P	J	1	P	J	1	P
K	0		K	1	P	K	1	P	K	1	R
A	0		A	0		A	1	Q	A	1	Q
B	0		B	0		B	1	Q	B	1	R
S	0		S	0		S	1	Q	S	1	R
C	0		C	0		C	0		C	1	R
T	0		T	0		T	0		T	1	R

28

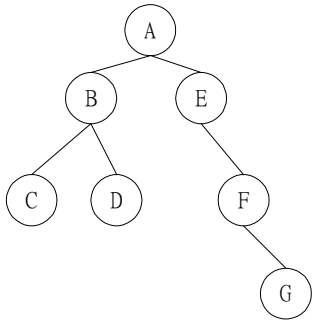
動態領域法範例：淺存取隱藏堆疊

<table>
<tr><td rowspan="4">Empty</td><td rowspan="4">Empty</td><td rowspan="4">x</td><td rowspan="4">main</td><td>x</td><td>Main</td></tr>
<tr><td>K</td><td>P</td></tr>
<tr><td>B</td><td>Q</td></tr>
<tr><td>s</td><td>Q</td></tr>
</table>

Main　　Main→ P　　Main→ P → Q　　Main→ P → Q → R

精選習題

- 假設區塊結構中的靜態巢狀樹如下所示：則



(a)在B的程式段中

(b)在F的程式段中

可以呼叫那些程式單元？

(a) A,B,C,D

(b) A,B,E,F,G

精選習題

- 一個變數(variable)，觀念上可分為以下六個成份：變數名稱(name) 、所佔空間地址、此空間可儲存資料的型態(type) 、此空間所儲存的值(value) 、其領域(scope) 、及其生命期間(life time)。

(1)試解釋在C的程序執行時，在什麼狀況下同一個程序的同一個變數名稱會代表不同的空間地址；而在什麼狀況下，不同的變數名稱會代表同空間地址。

(2)試簡述以下語言變數的生命週期(其佔有的空間，何時開始存在，何時停止存在)。Pascal local variables，C的static variables，Pascal的pointer variable所指的空間。

31