Computer Science – Lecture 14
Pascal Programming VI

Karl R. Wilcox
K.R.Wilcox@reading.ac.uk
Blackboard IFP

Objectives

- To talk about iteration statements
  - Doing things more than once
- (The practical sheet also covers formatted output)
  - (We will cover this topic next week)
- Tomorrow’s practical:
  - More Pascal Programming!

Iteration Statements

- Our first programs executed in order, from the first statement to the last
- Last week we looked at conditional statements
  - Executing one or more statements based on the result of a test expression
- Today we will look at executing statements more than once

Iteration Statements – For loop

```
var LoopCount: integer; Character: char;
begin
     for LoopCount := 1 to 5 do
        write (‘Loop ‘, LoopCount );
     writeln;

     for Character := ‘Z’ downto ‘V’ do
        write ( Character );
     writeln;
end.
```

For Loops – Points to Note

- The control variable cannot be of type "Real"
- Do NOT change the value of the control variable in the for loop
- Do NOT use the value of the control variable outside the for loop
- It is best to use "Begin" and "End" around the statement
  - Same reasons as with IF statements

Nested For Loops

- As with IF statements we can put one for loop inside another

```
var Outer, Inner: integer;

begin
  for Outer := 1 to 3 do
      for Inner := 6 to 7 do
      begin
          write (Inner); write (Outer);
      end
end
```

Initial and Final Values

- In the examples so far the initial and final values have been constants
- They can also be expressions
  - `for Count := Start + 100 to Size * 2 do`
- These expressions are evaluated once, when the loop is started
  - E.g. changing Start and Size inside the loop has no effect
- If the final value is greater than the initial value
  - The loop is never executed
- If the final value is the same as the initial value
  - The loop is executed once

When To Use For Loops

- When you know in advance exactly how many times you want to execute the code
  - e.g. to put 10 blank lines at the top of a page
    - `For Count := 1 to 10 do writeln;`

- When you can calculate in advance exactly how many times you want to execute the code
  - e.g. to print a line for each student

```
For Count := 1 to NumberOfStudents do
    writeln ( ‘present / absent’ );
```

Another Type of Loop - While

```
var Number: integer;

begin
      read ( Number );
      while Number < 10 do
         begin
            writeln (‘Option: ‘, Number );
            read ( Number );
         end
end.
```

While Loops – Points To Note

- If the expression evaluates to FALSE the first time
  - the loop will never be executed
- Loops may never end(!)
- One or more of the variables in the expression <u>should</u> be modified inside the loop
- While loops can be nested
  - For loops can be nested inside while loops
  - While loops can be nested inside for loops

When To Use While Loops

- When you do not know how many times you will need to execute the loops
  - E.g. when user actions control the loop
- When actions inside the loop control whether the loop should be run again
  - E.g. reading lines of text from a file
  - E.g. complex calculations, such as finding a square root by Newton's method

Summary

- Iteration Statements allow code to be executed more than once

- Remember:
  - Brackets are preferred around any expressions for readability
  - Use begin and end to group statements
    - (or for readability)
  - Iteration statements can be nested to any depth

Tomorrow’s Practical

- Try to write some of the programs suggested on the worksheet

- Worksheets available today if required
- Worksheets will take next 2 to 3 weeks to complete

- Karl will be available to help between 13:45 and 15:00 in the IT Degree Lab