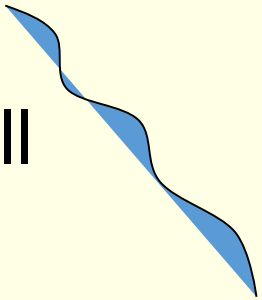
Anglia Ruskin University

- **Pseudocode**
- **Simple Debugging**
- **Iteration (using loops)**
- **Arrays**

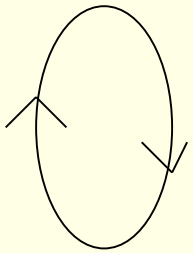
Anglia Ruskin University

The Software Life Cycle

Waterfall model



1. Requirements
2. Design
3. Implementation
4. Testing
5. Maintenance



Spiral model

Anglia Ruskin University

Pseudocode

Used for algorithm design,
i.e. creating a recipe

Example>

Write a program that calculates the mowing time in minutes and seconds for a rectangular block of land whose dimensions are input by the user. The mower mows at 2 square metres / second.

Anglia Ruskin University

Level 1 pseudocode

input width and length
area = width x length
time = area / rate
display time

Level 2 pseudocode

input width to nearest metre
input length to nearest metre
area = width x length
time = area / 2 rounded down to nearest second
mins = time / 60
secs = time % 60
display mins and secs

Anglia Ruskin University

C# implementation

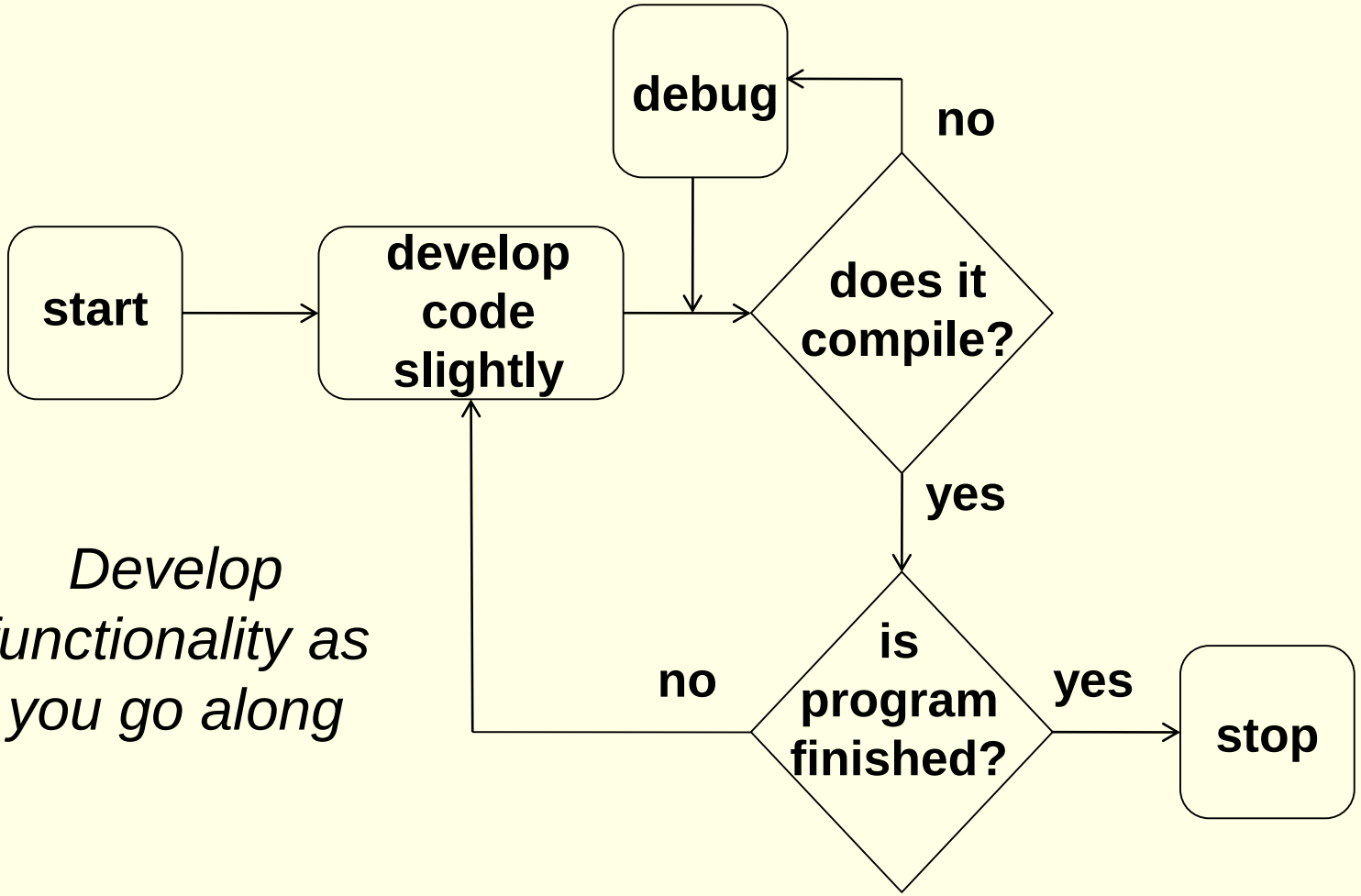
```
static void Main(string[ ] args)
{
    int width,  length;
    string input;
    Console.WriteLine("Enter width to nearest whole metre >");
    input = Console.ReadLine();
    width = int.Parse( input );
    Console.WriteLine("Enter length to nearest whole metre >");
    input = Console.ReadLine();
    length = int.Parse( input );
    int area = width * length;
    int time = area / 2;
    int mins = time / 60;
    int secs = time % 60;
    Console.WriteLine(“Time  is " + mins + “min “ + secs + “s” );
}
```

Anglia Ruskin University

- No accepted protocol as to how to write pseudocode
- No indication of type
- Step-wise (top-down) refinement can selectively clarify and identify possible devolvement to subprogram(s)
- Syntax common to many languages used eg ‘*’, ‘{‘

Anglia Ruskin University

Debugging



Develop functionality as you go along

Anglia Ruskin University

Develop functionality as you go along

- Trace statements

  Insert print commands at points to display values

  Console.WriteLine(“val is “ + num);
  Console.WriteLine(“ OK to here”);

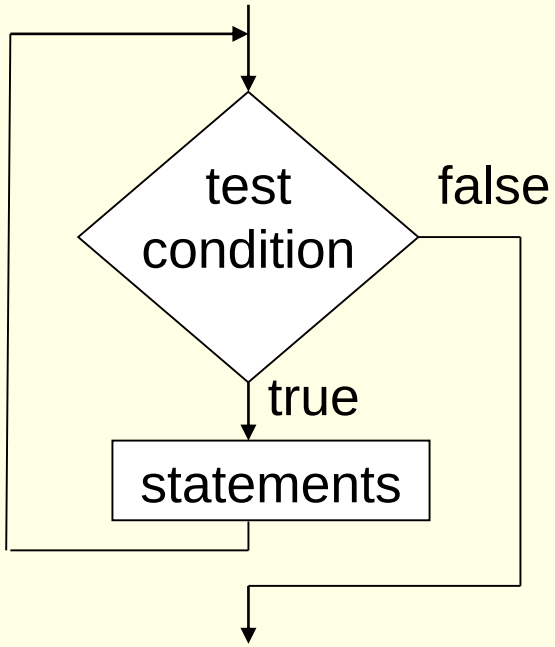
- Use the IDE Debugging tool

  Step-through the program line-by-line

Anglia Ruskin University

ITERATION

ITERATION: while, do-while, for

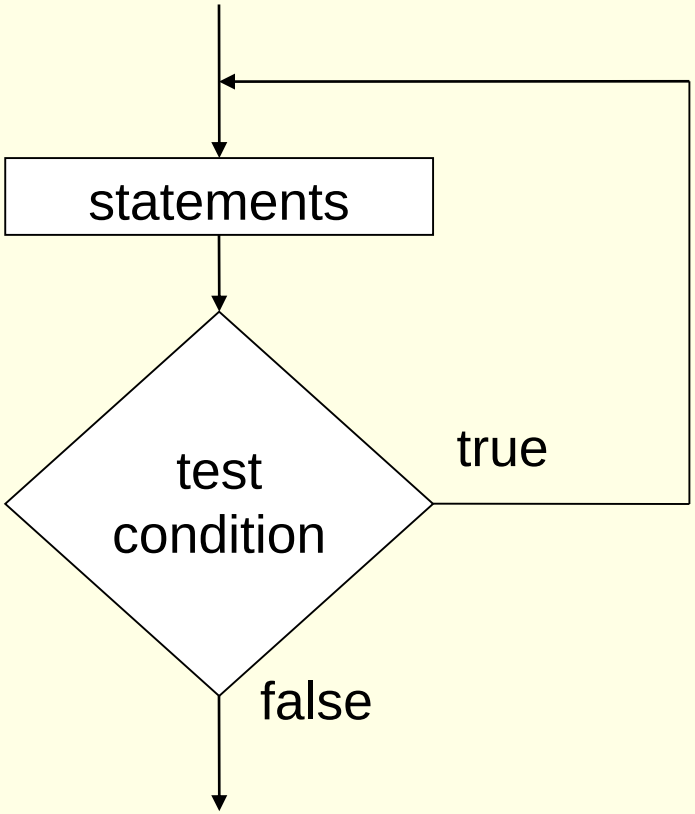


The while loop

```
int val = 4;
while ( val < 100 )
{
   val = val + 1;
}
```

Anglia Ruskin University

The do-while loop




```
int val = 4;
do
{
   val = val + 1;
} while ( val < 100 );
```

Anglia Ruskin University

The for loop

```
for( count=0; condition; count++ )
{
   statements;
}
```

```
double val = 3.4;
for( int num=0; num < 50; num++ )
{
   val = val * 1.5;
}
```

Post - incrementation

num++; *is short-hand for:* **num = num + 1;**

```
int  a, var = 6;
a = var++;      /* var= 7 and a=7 */
```

Anglia Ruskin University

Arrays

Primitive types: e.g. **int, double, bool,** etc

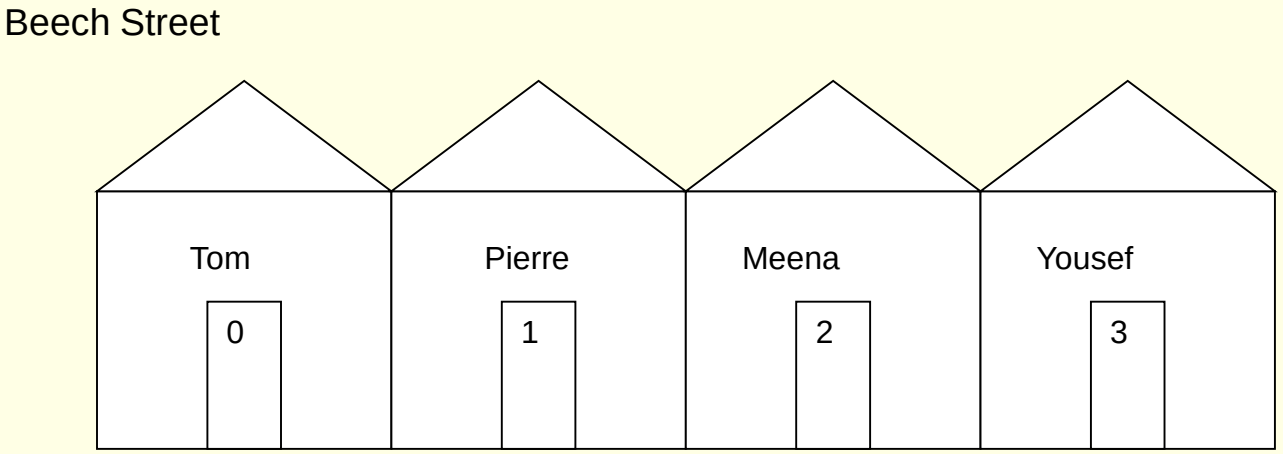
```
int val;
val = 6;
```

Can also define a collection of values of the same type; this is an *array*.

```
int[ ] numbers = { 4, 3, 9, 2, 5 };
```

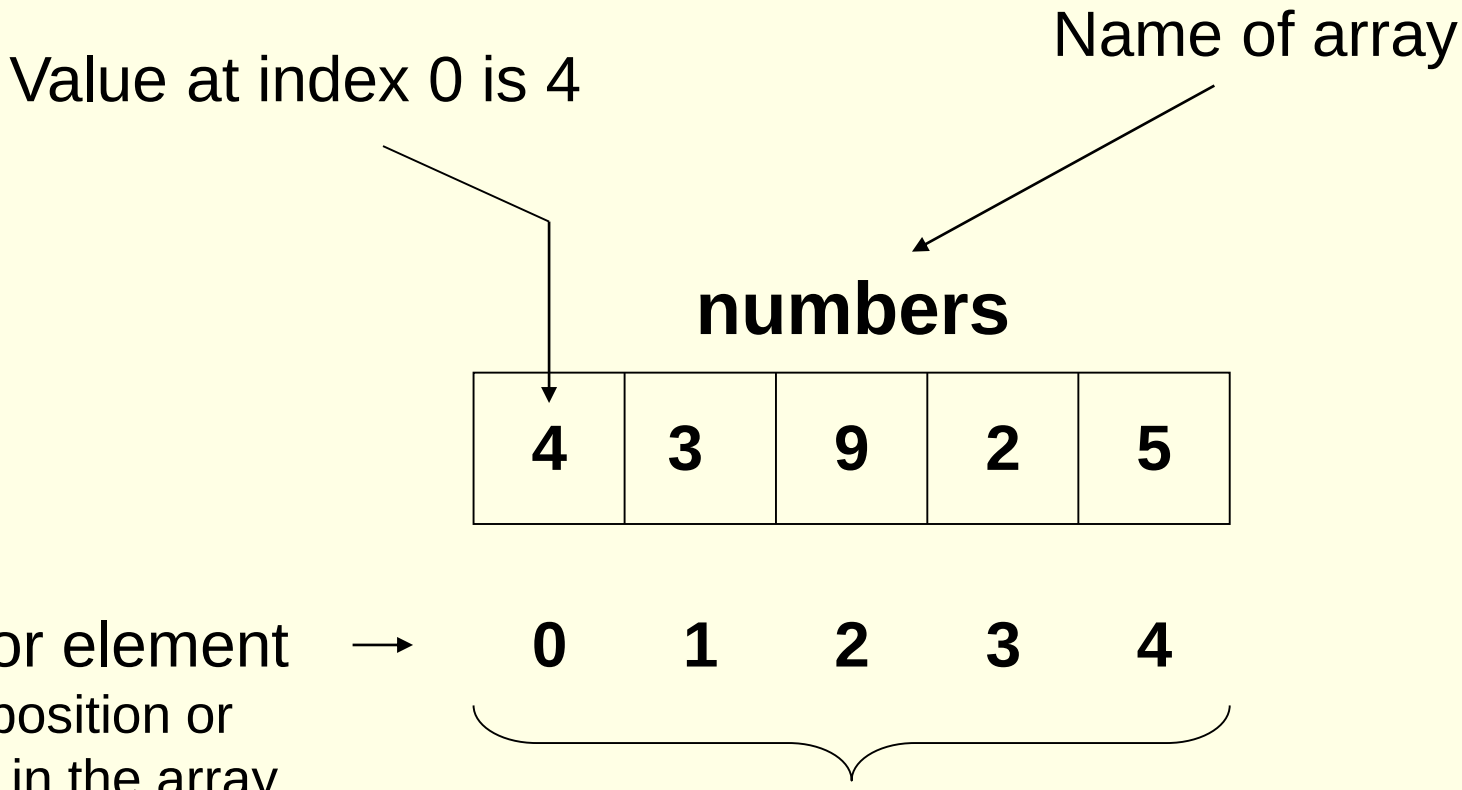
Anglia Ruskin University

An array is rather like a street of terraced houses



Name of the street = variable name of the array
Consecutive house numbers = elements of the array
Occupant of each house = value stored

Anglia Ruskin University



Length of array = 5
Elements 0 – 4

numbers.Length; //returns array length

Anglia Ruskin University

```
int  value = numbers[ 3 ];
Console.WriteLine( “Value at index 3 is “ + value );
```

Value at index 3 is 2

Alternatively could use a loop

```
for (int index=0; index < numbers.Length; index++ )
{
   Console.WriteLine( “Value at “+ index +“ is “+numbers[index] );
}
```

Value at 0 is 4
Value at 1 is 3
Value at 2 is 9
Value at 3 is 2
Value at 4 is 5

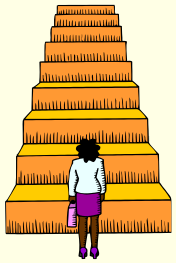
Variation is to loop through array using **foreach**

Anglia Ruskin University

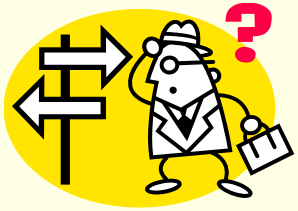
Concluding thought

- Loops are often used to process arrays
- Control structures determine the flow of execution

- sequence



- selection



- repetition

ANY program can be written using **sequential operations**, **if**, and **while** – the other structures are variations to make coding easier