Records (Revision)

Let's look at a more efficient way

For our BMI program we were storing 3 values in 3 parallel arrays:

1. Height
2. Weight
3. BMI

We can store these in one record structure where each one will have these 3 pieces of data

Defining a record in Pseudocode

RECORD BMI:

Height : REAL

Weight: REAL

BMI: REAL

END RECORD

This defines a data structure that has 3 pieces of information.

But we will need to have an array of these as we will be holding details about numerous people so...

Using a record in Pseudocode

If we need to declare a single BMI variable

- `DECLARE BMIrecord AS BMI`

And then to use the different ‘fields’ we can use the following:

- `BMIrecord.height  = 1.6`
- `BMIrecord.weight  = 62.3`
- `BMIrecord.BMI = BMIrecord.weight * (BMIrecord.height ^ 2)`

Let's look at a single record in Python...

```
class BMI():
    def __init__(self):
        self.height = 0.0
        self.weight = 0.0
        self.BMI = 0.0

myBMI = BMI()
myBMI.height = 1.6
myBMI.weight = 62.3
myBMI.BMI = myBMI.weight / myBMI.height **2

print (myBMI.BMI)
```

Declares a class called **BMI** (not officially a record but is what we will use in Python

Declares a **single BMI Record**

And then assigns the values in the relevant fields and then calculates the BMI

Displays the BMI we just calculated

But we will probably need more than one

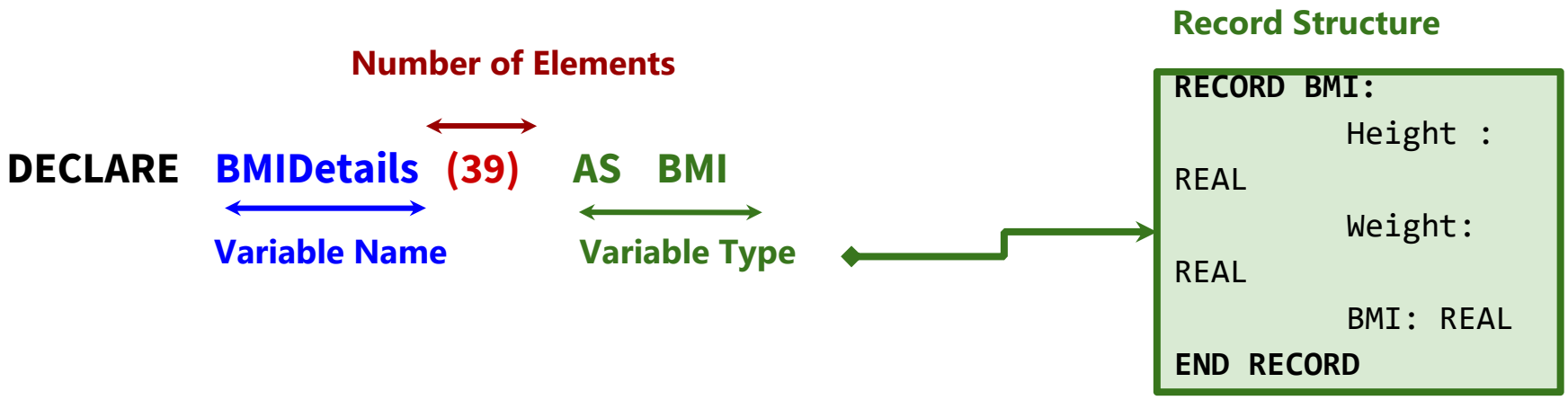
As we want to store numerous records such as in the table shown below:

We will need to declare an **array** of records to store each of the set of fields

Record Number	Height	Weight	BMI
0	1.6	62.3	24.3
1	1.7	73.3	25.4
2	1.3	44.0	26.0
3	1.4	58.1	29.6

Declaring an array of records (EXAM)

Python doesn't require you to declare variables with types explicitly as it is a loosely typed language so in an exam situation if you were asked to store 40 BMI records you could use:



Using an array of records in Pseudocode

```
DECLARE BMIDetails (39) AS BMI

BMIDetails[0].height = 1.6

BMIDetails[0].weight = 62.3
```

This stores the following:

Record Number	Height	Weight	BMI
0	1.6	62.3	
1			

Using an array of records in Pseudocode (cont)

```
BMIDetails[0].BMI = BMIDetails[0].weight / (BMIDetails[0].height ^ 2)
```

We have used the height and weight fields to calculate the BMI for the first record so are now storing all of the data shown below:

Record Number	Height	Weight	BMI
0	1.6	62.3	24.3
1			

Next person(s)...

```
BMIDetails[1].height = 1.7

BMIDetails[1].weight = 73.3

BMIDetails[1].BMI = BMIDetails[1].weight / (BMIDetails[1].height ^ 2)
```

Record Number	Height	Weight	BMI
0	1.6	62.3	24.3
1	1.7	73.3	25.4

And multiple ones (without a loop)

```
class BMI():
    def __init__(self):
        self.height = 0.0
        self.weight = 0.0
        self.BMI = 0.0

myBMI = [BMI() for x in range (40)]

myBMI[0].height = 1.6
myBMI[0].weight = 62.3
myBMI[0].BMI = myBMI[0].weight / myBMI[0].height **2

print (myBMI[0].BMI)
```

Declares a class called **BMI** (not officially a record but is what we will use in Python

Declares an **array of 40 BMI Records**

And then assigns the values to the fields for the first records in **element 0** of the **myBMI Array**

Displays the first BMI we just calculated

It would be more efficient to use a loop

```
DECLARE BMIDetails (39) AS BMI

FOR i = 0 to LENGTH(BMIDetails)

BMIDetails[i].height = GET INPUT FROM USER

BMIDetails[i].weight = GET INPUT FROM USER

BMIDetails[i].BMI = BMIDetails[i].height / (BMIDetails[i].weight ^ 2)

NEXT LOOP
```

And multiple records in Python (with a loop)

```
class BMI():
    def __init__(self):
        self.height = 0.0
        self.weight = 0.0
        self.BMI = 0.0

myBMI = [BMI() for x in range (40)]

for x in range(len(myBMI)):
    myBMI[x].height = float(input("Enter Height: "))
    myBMI[x].weight = float(input("Enter Weight: "))
    myBMI[x].BMI = roundmyBMI[x].weight / myBMI[x].height **2
    print ("BMI = ", myBMI[x].BMI)
```

Loops through for the length of the array

Uses the loop counter [x] to place the values at the specified elements in the array