PSEUDOCODES & ACTIVITIES

THINGS TO DO TODAY

GRAD 10 STUDENTS

You are the log-in to HTTPS://IDEA.Org.Uk, and get the BRONZE, SILVER, or GOLD awards by completing ICT/CS related badges

IGCSE CS 0478 CANDIDATES

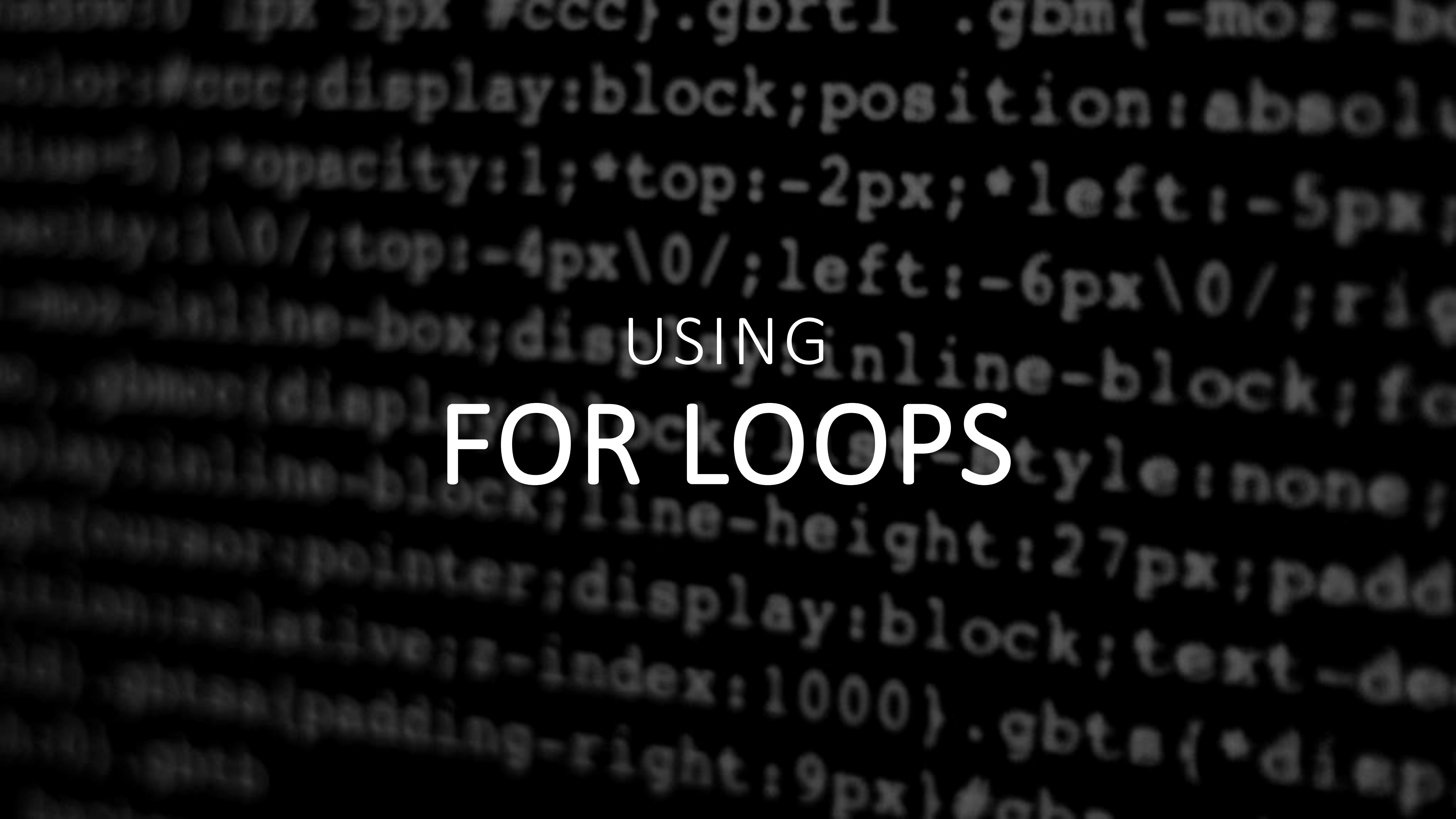
Take the front row and pay attention with the activities for PSEUDOCODEs.

LOOPS IN PSEUDOCODES

Loops are used to repeat a set of instructions some number of times. This is a programming concept known as iteration or repetition.

In the python programming language, there are two different loop structures that we can use. One structure is a for loop, and the other structure is a while loop.

When writing pseudocode, we can still use **FOR loops and WHILE loops**, but there is also a third loop structure using the **REPEAT command**.



USING

FOR LOOPS

For loops

Example 1

```
for x in range(10):
    print(x)
```

Example 2

```
Num = int(input())
for x in range(1,11):
    print(Num * x)
```

Example 3

```
Sum = 0
for x in range(5):
    Num = int(input())
    Sum = Sum + Num
print(Sum)
```

For loops

Example 1

```
for x in range(10):
   print(x)
```

Example 1

```
FOR X ← 0 TO 9
   PRINT X
NEXT X
```

Example 2

```
Num = int(input())
for x in range(1,11):
   print(Num * x)
```

Example 3

```
Sum = 0
for x in range(5):
   Num = int(input())
   Sum = Sum + Num
print(Sum)
```

For loops

Example 1

```
for x in range(10):
   print(x)
```

Example 1

```
FOR X ← 0 TO 9
   PRINT X
NEXT X
```

Example 2

```
Num = int(input())
for x in range(1,11):
   print(Num * x)
```

Example 2

```
INPUT Num
FOR X ← 1 TO 10
   PRINT Num * X
NEXT X
```

Example 3

```
Sum = 0
for x in range(5):
   Num = int(input())
   Sum = Sum + Num
print(Sum)
```

For loops

Example 1

```
for x in range(10):
    print(x)
```

Example 2

```
Num = int(input())
for x in range(1,11):
    print(Num * x)
```

Example 3

```
Sum = 0
for x in range(5):
    Num = int(input())
    Sum = Sum + Num
print(Sum)
```

Example 1

```
FOR X ← 0 TO 9
    PRINT X
NEXT X
```

Example 2

```
INPUT Num
FOR X ← 1 TO 10
    PRINT Num * X
NEXT X
```

Example 3

```
Sum ← 0
FOR X ← 1 TO 5
    INPUT Num
    Sum ← Sum + Num
NEXT X
PRINT Sum
```

For loops

-
  ```
  for x in range(5):
      print(x)
  ```

-
  ```
  for x in range(1,4):
      Num = int(input())
      print(Num ** 2)
  ```

-
  ```
  for Count in range(100):
      print(Values[Count])
  ```

-
  ```
  Product = 1
  for i in range(1,5):
      Num = int(input())
      Product = Product * Num
  print(Product)
  ```

-
  ```
  for Count in range(20):
   Number = int(input())
   A[Count] = Number
  print(A)
  ```

-
  ```
  Max = int(input())
  for Count in range(Max):
     print(Count)
  ```

For loops

-
  ```
  for x in range(5):
      print(x)
  ```

-
  ```
  for x in range(1,4):
      Num = int(input())
      print(Num ** 2)
  ```

-
  ```
  for Count in range(100):
      print(Values[Count])
  ```

-
  ```
  Product = 1
  for i in range(1,5):
      Num = int(input())
      Product = Product * Num
  print(Product)
  ```

-
  ```
  for Count in range(20):
   Number = int(input())
   A[Count] = Number
  print(A)
  ```

-
  ```
  Max = int(input())
  for Count in range(Max):
      print(Count)
  ```

(a)
```
FOR x ← 0 TO 4
  PRINT x
NEXT x
```

(b)
```
FOR x ← 1 TO 4
    INPUT Num
    PRINT Num ^ 2
NEXT x
```

(c)
```
FOR Count ← 1 TO 100
  PRINT Values[Count]
NEXT Count
```

(d)
```
Product ← 1
   FOR i ← 1 TO 4
   INPUT Num
   Product ← Product * Num
   NEXT i
 PRINT Product
```

(e)
```
FOR Count ← 1 TO 20
   INPUT Number
   A[Count] ← Number
   NEXT Count
PRINT A
```

(f)
```
INPUT Max
   FOR Count ← 1 TO Max
   PRINT Count
NEXT Count
```

For loops

-
  ```
  for x in range(5):
      print(x)
  ```

-
  ```
  for x in range(1,4):
      Num = int(input())
      print(Num ** 2)
  ```

-
  ```
  for Count in range(100):
      print(Values[Count])
  ```

-
  ```
  Product = 1
  for i in range(1,5):
      Num = int(input())
      Product = Product * Num
  print(Product)
  ```

-
  ```
  for Count in range(20):
   Number = int(input())
   A[Count] = Number
  print(A)
  ```

-
  ```
  Max = int(input())
  for Count in range(Max):
     print(Count)
  ```

(a)
```
FOR x ← 0 TO 4
  PRINT x
NEXT x
```

(b)
```
FOR x ← 1 TO 4
   INPUT Num
   PRINT Num ^ 2
NEXT x
```

(c)
```
FOR Count ← 0 TO 99
  PRINT Values[Count]
NEXT Count
```

(d)
```
Product ← 1
   FOR i ← 1 TO 5
   INPUT Num
   Product ← Product * Num
   NEXT i
 PRINT Product
```

(e)
```
FOR Count ← 1 TO 20
  INPUT Number
  A[Count] ← Number
  NEXT Count
PRINT A
```

(f)
```
INPUT Max
   FOR Count ← 1 TO Max
   PRINT Count
NEXT Count
```

For loops

-
  ```
  for x in range(5):
      print(x)
  ```

-
  ```
  for x in range(1,4):
      Num = int(input())
      print(Num ** 2)
  ```

-
  ```
  for Count in range(100):
      print(Values[Count])
  ```

-
  ```
  Product = 1
  for i in range(1,5):
      Num = int(input())
      Product = Product * Num
  print(Product)
  ```

-
  ```
  for Count in range(20):
   Number = int(input())
   A[Count] = Number
  print(A)
  ```

-
  ```
  Max = int(input())
  for Count in range(Max):
     print(Count)
  ```

(a)
```
FOR x ← 0 TO 4
  PRINT x
NEXT x
```

(b)
```
FOR x ← 1 TO 4
  INPUT Num
  PRINT Num ^ 2
NEXT x
```

(c)
```
FOR Count ← 0 TO 99
 PRINT Values[Count]
NEXT Count
```

(d)
```
Product ← 1
  FOR i ← 1 TO 5
  INPUT Num
  Product ← Product * Num
  NEXT i
PRINT Product
```

(e)
```
FOR Count ← 1 TO 20
  INPUT Number
  A[Count] ← Number
  NEXT Count
PRINT A
```

(f)
```
INPUT Max
  FOR Count ← 1 TO Max
  PRINT Count
NEXT Count
```

For loops

-
  ```
  for x in range(5):
      print(x)
  ```

-
  ```
  for x in range(1,4):
      Num = int(input())
      print(Num ** 2)
  ```

-
  ```
  for Count in range(100):
      print(Values[Count])
  ```

-
  ```
  Product = 1
  for i in range(1,5):
      Num = int(input())
      Product = Product * Num
  print(Product)
  ```

-
  ```
  for Count in range(20):
   Number = int(input())
   A[Count] = Number
  print(A)
  ```

-
  ```
  Max = int(input())
  for Count in range(Max):
      print(Count)
  ```

(a)
```
FOR x ← 1 TO 5
  PRINT x
NEXT x
```

(b)
```
FOR x ← 1 TO 4
   INPUT Num
   PRINT Num ^ 2
NEXT x
```

(c)
```
FOR Count ← 1 TO 100
  PRINT Values[Count]
NEXT Count
```

(d)
```
Product ← 1
   FOR i ← 1 TO 5
   INPUT Num
   Product ← Product * Num
   NEXT i
 PRINT Product
```

(e)
```
FOR Count ← 1 TO 20
   INPUT Number
   A[Count] ← Number
   NEXT Count
PRINT A
```

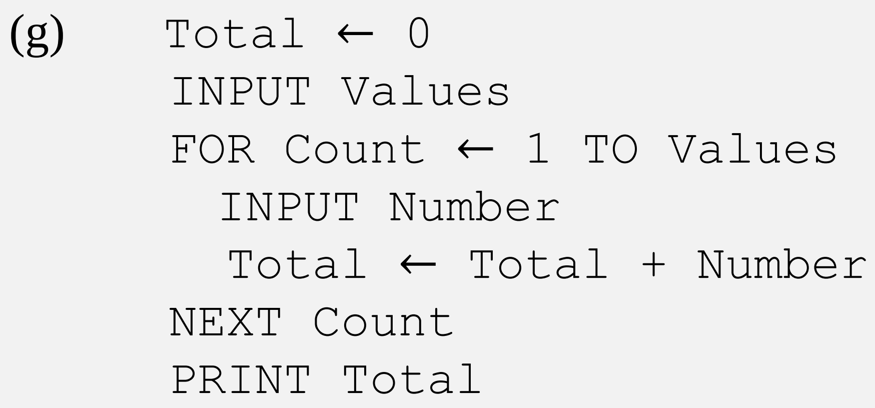
(f)
```
INPUT Max
   FOR Count ← 1 TO Max
   PRINT Count
NEXT Count
```

For loops

```
Total = 0
Values = int(input())
for Count in range(Values):
    Number = int(input())
    Total = Total + Number
print(Total)
```

For loops

-
  ```
  Total = 0
  Values = int(input())
  for Count in range(Values):
      Number = int(input())
      Total = Total + Number
  print(Total)
  ```



(g)

```
Total ← 0
INPUT Values
FOR Count ← 1 TO Values
   INPUT Number
   Total ← Total + Number
NEXT Count
PRINT Total
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a FOR loop, to print "Hello" five times.

2. Write an algorithm, using pseudocode with a FOR loop, to print the numbers from 1 to 20.

3. Write an algorithm, using pseudocode containing a FOR loop, to:
   - ask the user to enter 50 numbers
   - work out the total of the numbers entered, and output the result
   - work out the average of the numbers entered, and output the result

4. The 1D array Heights contains the heights of 25 students in a class.
   - Write an algorithm, using pseudocode containing a FOR loop, to find and output the height of the tallest student in the class.

Write down your PSEUDOCODEs
Follow the Cambridge IGCSE 0478 Standards

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a FOR loop, to print "Hello" five times.

```
DECLARE Count : INTEGER
FOR Count ← 1 TO 5
   PRINT "Hello"
NEXT Count
```

2. Write an algorithm, using pseudocode with a FOR loop, to print the numbers from 1 to 20.

```
DECLARE Count : INTEGER
FOR Count ← 1 TO 20
   PRINT Count
NEXT Count
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

3. Write an algorithm, using pseudocode containing a `FOR` loop, to:
   - ask the user to enter 50 numbers
   - work out the total of the numbers entered, and output the result
   - work out the average of the numbers entered, and output the result

4. The 1D array `Heights` contains the heights of 25 students in a class.
   - Write an algorithm, using pseudocode containing a `FOR` loop, to find and output the height of the tallest student in the class.

Write down your PSEUDOCODEs

Follow the Cambridge IGCSE 0478 Standards

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

3. Write an algorithm, using pseudocode containing a `FOR` loop, to:
   - ask the user to enter 50 numbers
   - work out the total of the numbers entered, and output the result
   - work out the average of the numbers entered, and output the result

```
DECLARE Num : INTEGER
DECLARE Total : INTEGER
DECLARE Count : INTEGER
Total ← 0
FOR Count ← 1 TO 50
    INPUT Num
    Total ← Total + Num
NEXT Count
PRINT TOTAL
PRINT Total / 50
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. The 1D array `Heights` contains the heights of 25 students in a class.
    - Write an algorithm, using pseudocode containing a `FOR` loop, to find and output the height of the tallest student in the class.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. The 1D array `Heights` contains the heights of 25 students in a class.
    - Write an algorithm, using pseudocode containing a `FOR` loop, to find and output the height of the tallest student in the class.

```
DECLARE Count : INTEGER
DECLARE Max : REAL
Max ← 0
FOR Count ← 1 TO 25
  IF Heights[Count] > Max
    THEN Max ← Heights[Count]
  ENDIF
NEXT Count
PRINT Max
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

3. Write an algorithm, using pseudocode containing a `FOR` loop, to:
    - ask the user to enter 50 numbers
    - work out the total of the numbers entered, and output the result
    - work out the average of the numbers entered, and output the result
    -

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

3. Write an algorithm, using pseudocode containing a `FOR` loop, to:
   - ask the user to enter 50 numbers
   - work out the total of the numbers entered, and output the result
   - work out the average of the numbers entered, and output the result

```
DECLARE Num : INTEGER
DECLARE Total : INTEGER
DECLARE Count : INTEGER
Total ← 0
FOR Count ← 1 TO 50
     INPUT Num
     Total ← Total + Num
NEXT Count
PRINT TOTAL
PRINT Total / 50
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. The 1D array `Heights` contains the heights of 25 students in a class.
    - Write an algorithm, using pseudocode containing a `FOR` loop, to find and output the height of the tallest student in the class.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. The 1D array `Heights` contains the heights of 25 students in a class.
    - Write an algorithm, using pseudocode containing a `FOR` loop, to find and output the height of the tallest student in the class.

```
DECLARE Count : INTEGER
DECLARE Max : Real
Max ← 0
FOR Count ← 1 TO 25
  IF Heights[Count] > Max
    THEN Max ← Heights[Count]
  ENDIF
NEXT Count
PRINT Max
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode with a FOR loop, to input 100 numbers into an array.

5. Write an algorithm, using pseudocode with a FOR loop, to output 30 names stored in an array.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode with a FOR loop, to input 100 numbers into an array.

5. Write an algorithm, using pseudocode with a FOR loop, to output 30 names stored in an array.

```
4. DECLARE A : ARRAY [1:100] OF REAL
   DECLARE Count : INTEGER
   FOR Count ← 1 TO 100
        INPUT A[Count]
   NEXT Count
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode with a FOR loop, to input 100 numbers into an array.

5. Write an algorithm, using pseudocode with a FOR loop, to output 30 names stored in an array.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode with a FOR loop, to input 100 numbers into an array.

5. Write an algorithm, using pseudocode with a FOR loop, to output 30 names stored in an array.

```
4.   DECLARE A : ARRAY [1:100] OF REAL
    DECLARE Count : INTEGER
    FOR Count ← 1 TO 100
         INPUT A[Count]
    NEXT Count


5.  DECLARE A : ARRAY [1:30] OF STRING
    DECLARE Count : INTEGER
    FOR Count ← 1 TO 30
         PRINT A[Count]
    NEXT Count
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a whole number output all the positive integers up to and including their number

7. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a number output the first three multiples of the number (e.g.first three multiples of 4 are 4, 8, 12)

For loops worded problems

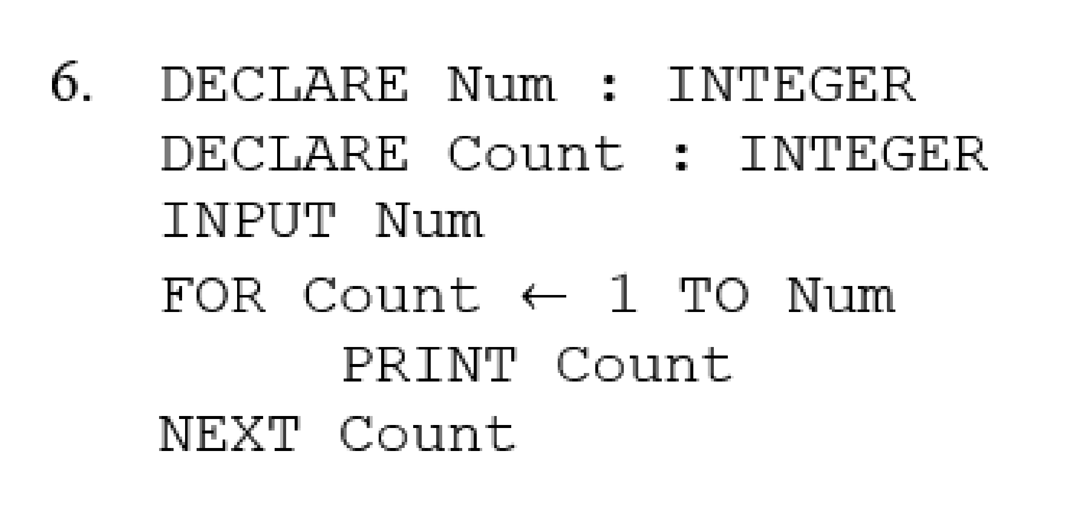
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a whole number output all the positive integers up to and including their number

7. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a number output the first three multiples of the number (e.g.first three multiples of 4 are 4, 8, 12)



```
6.  DECLARE Num : INTEGER
    DECLARE Count : INTEGER
    INPUT Num
    FOR Count ← 1 TO Num
         PRINT Count
    NEXT Count
```

For loops worded problems

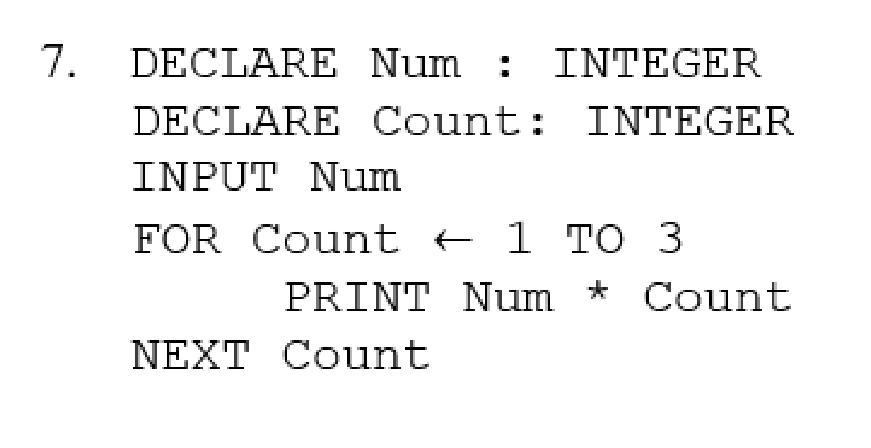
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a whole number output all the positive integers up to and including their number

7. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a number output the first three multiples of the number (e.g.first three multiples of 4 are 4, 8, 12)



```
7.  DECLARE Num : INTEGER
    DECLARE Count: INTEGER
    INPUT Num
    FOR Count ← 1 TO 3
         PRINT Num * Count
    NEXT Count
```

For loops worded problems

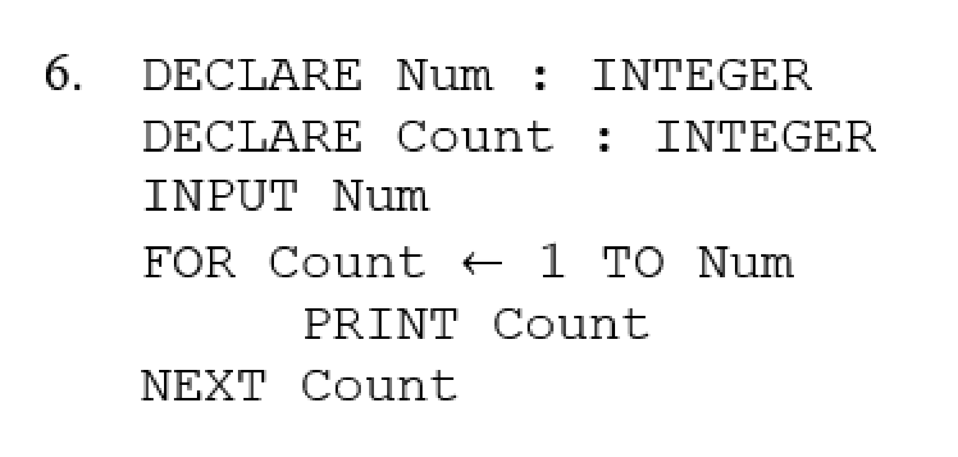
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a FOR loop, to:

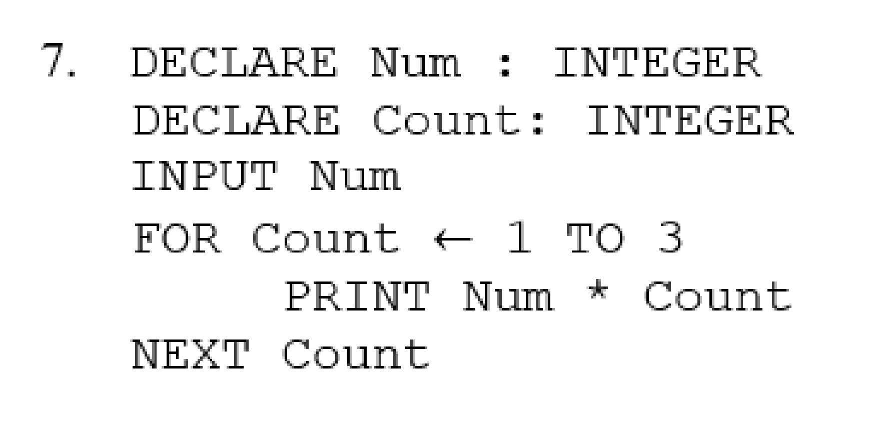
ask the user to enter a whole number output all the positive integers up to and including their number

7. Write an algorithm, using pseudocode containing a FOR loop, to:

ask the user to enter a number output the first three multiples of the number (e.g.first three multiples of 4 are 4, 8, 12)



```
6.  DECLARE Num : INTEGER
    DECLARE Count : INTEGER
    INPUT Num
    FOR Count ← 1 TO Num
          PRINT Count
    NEXT Count
```



```
7.  DECLARE Num : INTEGER
    DECLARE Count: INTEGER
    INPUT Num
    FOR Count ← 1 TO 3
          PRINT Num * Count
    NEXT Count
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

8. Write an algorithm, using pseudocode containing a FOR loop, to:

- ask the user to enter a whole number
- work out the first twelve multiples of the number, storing the results in an array

9. Write an algorithm, using pseudocode containing a FOR loop, to:

- ask the user to enter 50 numbers
- work out the total of the numbers entered, and output the result
- work out the average of the numbers entered, and output the result

For loops worded problems

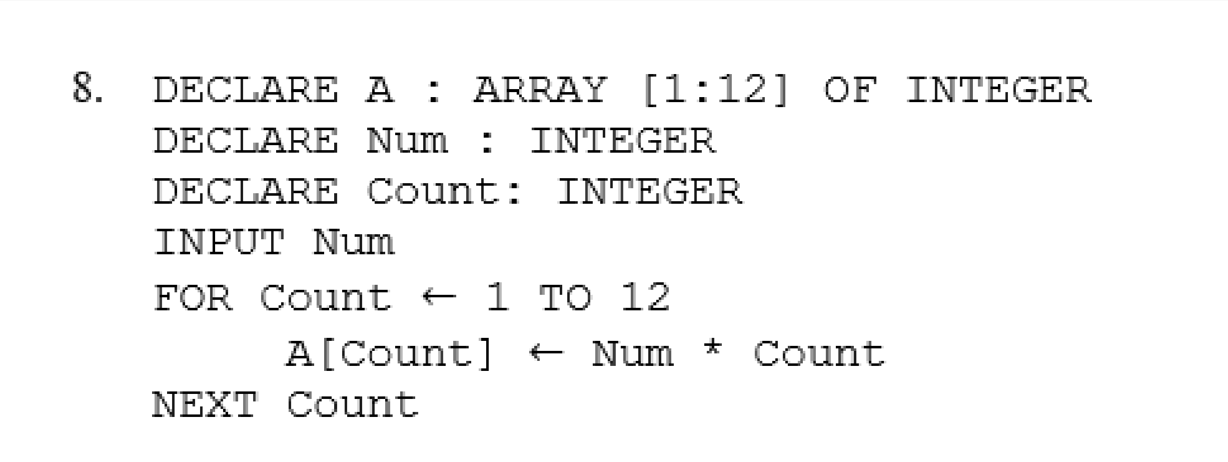
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

8. Write an algorithm, using pseudocode containing a FOR loop, to:

•ask the user to enter a whole number
•work out the first twelve multiples of the number, storing the results in an array

9. Write an algorithm, using pseudocode containing a FOR loop, to:

•ask the user to enter 50 numbers
•work out the total of the numbers entered, and output the result
•work out the average of the numbers entered, and output the result



```
8.  DECLARE A : ARRAY [1:12] OF INTEGER
    DECLARE Num : INTEGER
    DECLARE Count: INTEGER
    INPUT Num
    FOR Count ← 1 TO 12
         A[Count] ← Num * Count
    NEXT Count
```

For loops worded problems

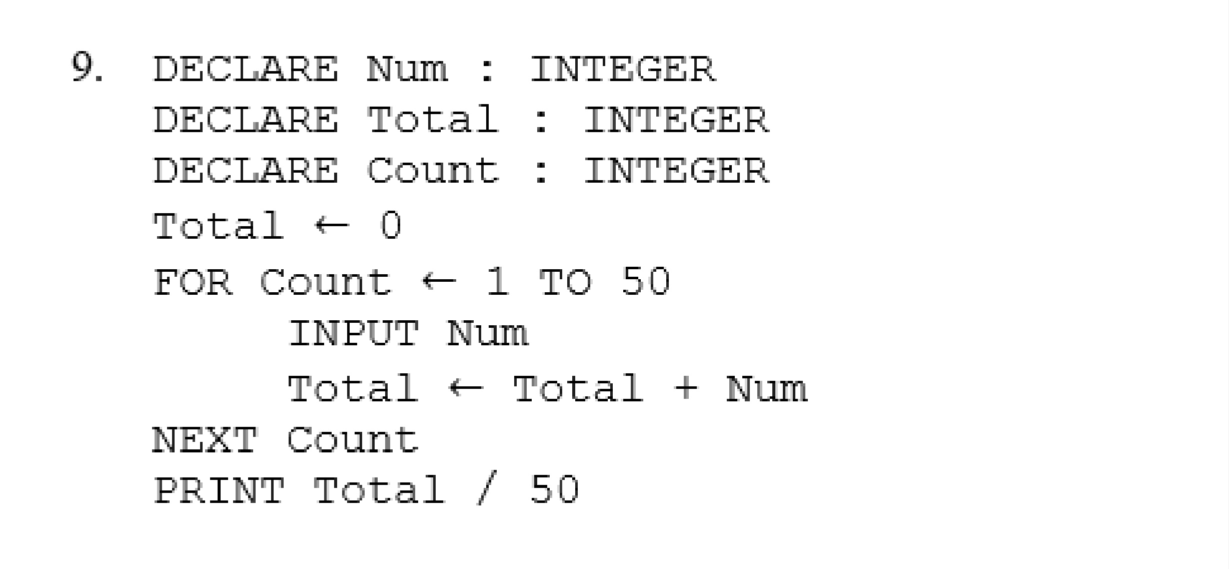
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

8. Write an algorithm, using pseudocode containing a FOR loop, to:

- ask the user to enter a whole number
- work out the first twelve multiples of the number, storing the results in an array

9. Write an algorithm, using pseudocode containing a FOR loop, to:

- ask the user to enter 50 numbers
- work out the total of the numbers entered, and output the result
- work out the average of the numbers entered, and output the result



```
9.  DECLARE Num : INTEGER
    DECLARE Total : INTEGER
    DECLARE Count : INTEGER
    Total ← 0
    FOR Count ← 1 TO 50
         INPUT Num
         Total ← Total + Num
    NEXT Count
    PRINT Total / 50
```

For loops worded problems

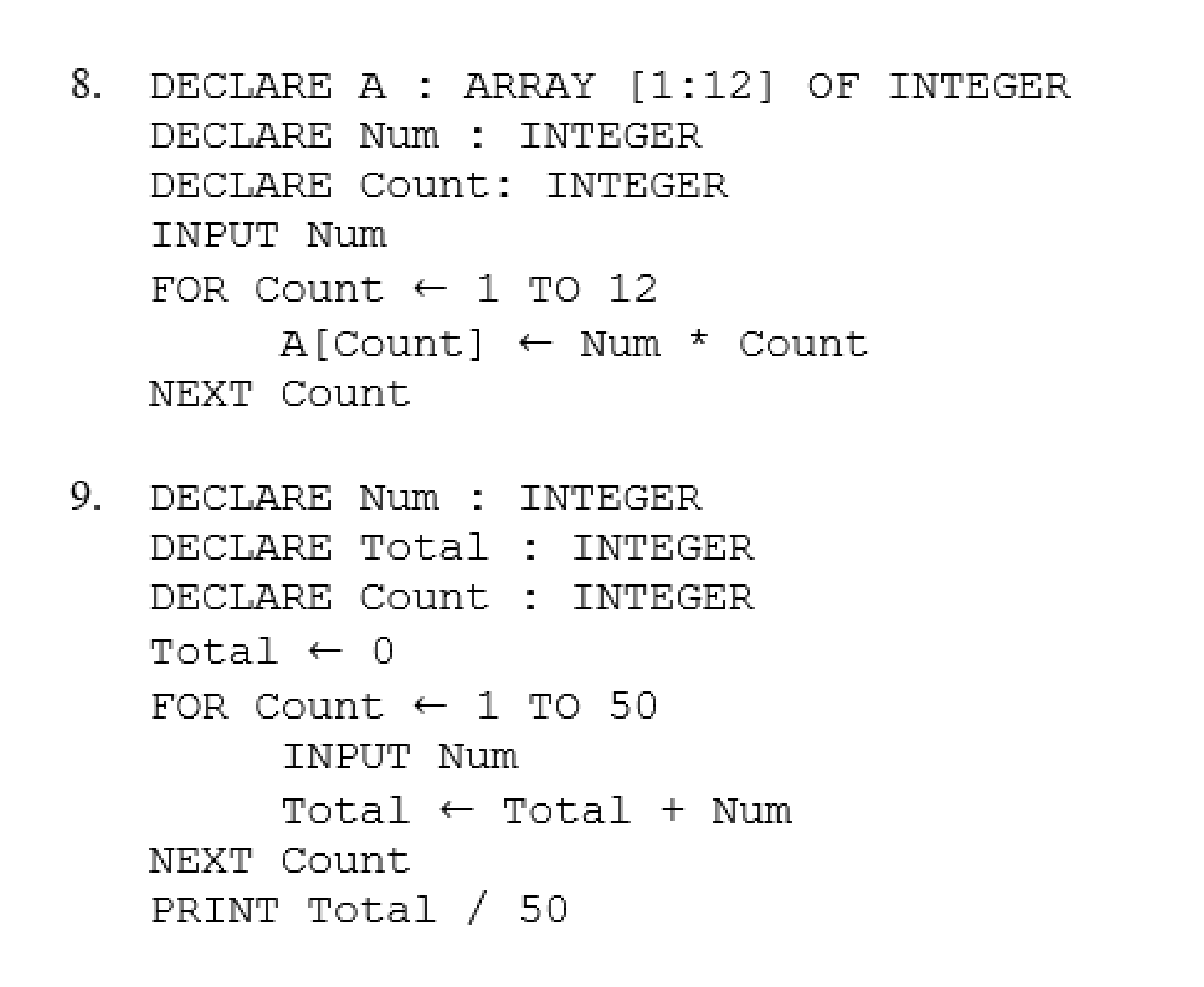
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

8. Write an algorithm, using pseudocode containing a FOR loop, to:

- ask the user to enter a whole number
- work out the first twelve multiples of the number, storing the results in an array

9. Write an algorithm, using pseudocode containing a FOR loop, to:

- ask the user to enter 50 numbers
- work out the total of the numbers entered, and output the result
- work out the average of the numbers entered, and output the result



```
8.  DECLARE A : ARRAY [1:12] OF INTEGER
    DECLARE Num : INTEGER
    DECLARE Count: INTEGER
    INPUT Num
    FOR Count ← 1 TO 12
         A[Count] ← Num * Count
    NEXT Count

9.  DECLARE Num : INTEGER
    DECLARE Total : INTEGER
    DECLARE Count : INTEGER
    Total ← 0
    FOR Count ← 1 TO 50
         INPUT Num
         Total ← Total + Num
    NEXT Count
    PRINT Total / 50
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

16. The 1D array Scores contains the results of 1000 students who entered a maths challenge. Any students with a score of 80 or more qualify to take part in the next round.

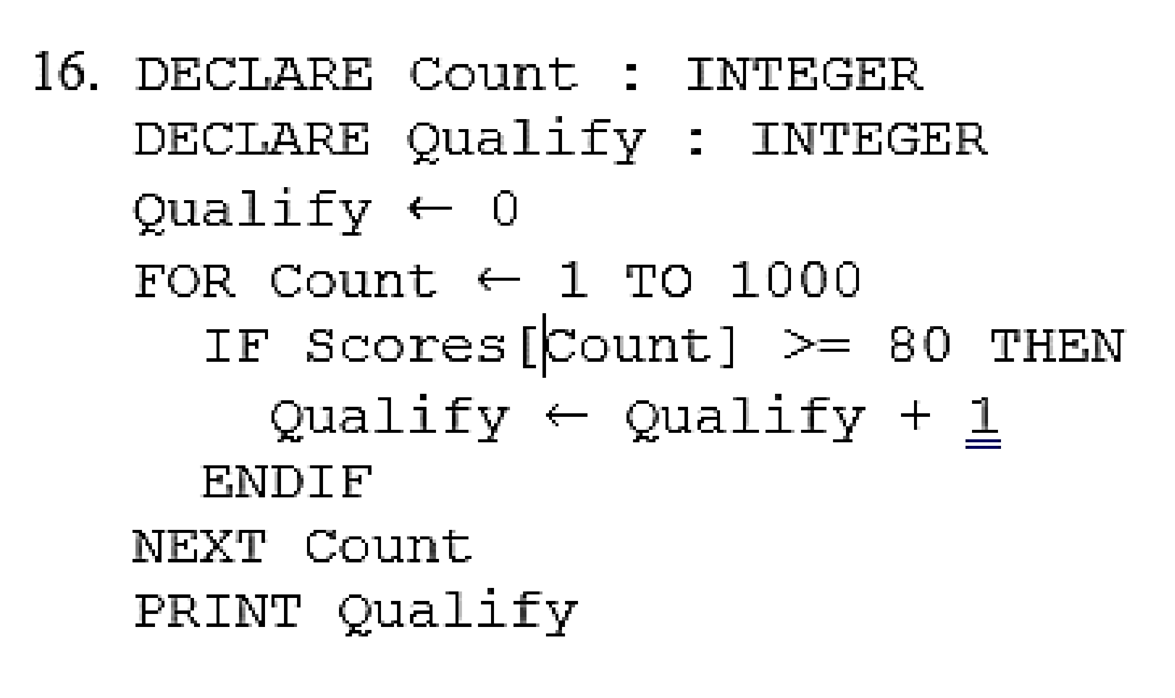
Write an algorithm, using pseudocode containing a FOR loop, to find and output the number of students who qualify for the next round.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

16. The 1D array Scores contains the results of 1000 students who entered a maths challenge. Any students with a score of 80 or more qualify to take part in the next round.

Write an algorithm, using pseudocode containing a FOR loop, to find and output the number of students who qualify for the next round.



```
16. DECLARE Count : INTEGER
    DECLARE Qualify : INTEGER
    Qualify ← 0
    FOR Count ← 1 TO 1000
      IF Scores[Count] >= 80 THEN
        Qualify ← Qualify + 1
      ENDIF
    NEXT Count
    PRINT Qualify
```

For loops worded problems

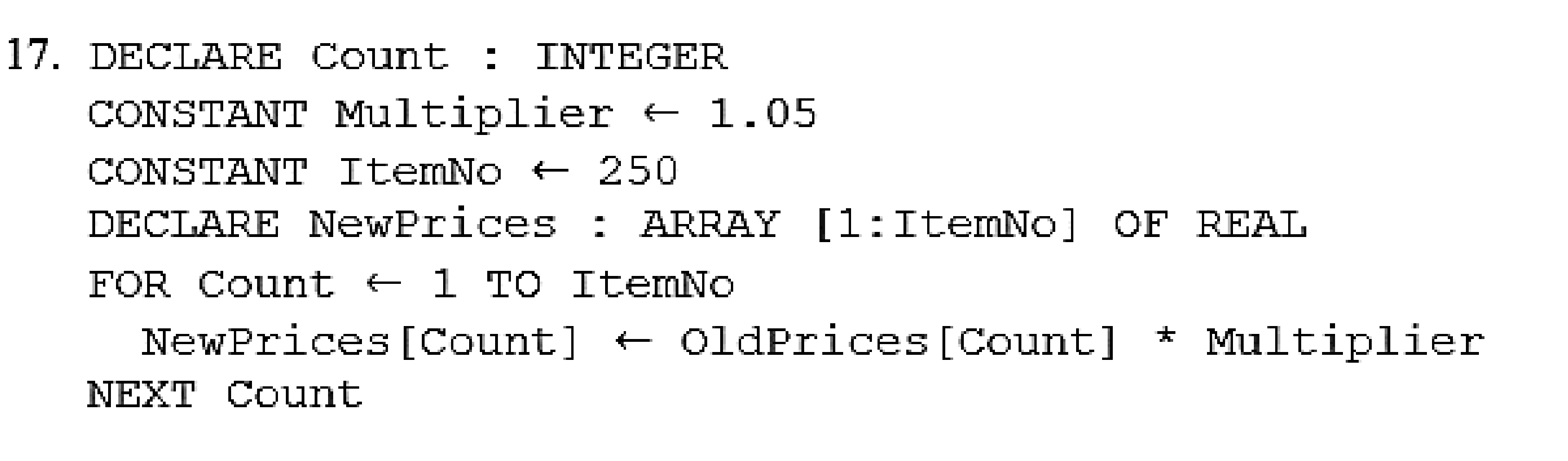
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

17. A store manager is thinking about prices of the 250 items in her store.
Last year's prices are recorded in a 1D array called OldPrices.
To work out each item's new price, the manager needs to multiply its price from last year by the number 1.05
The variable ItemNo stores the number of items for sale in her store.
Write an algorithm, using pseudocode containing a FOR loop, to calculate the new price for each item, storing the prices in an array.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

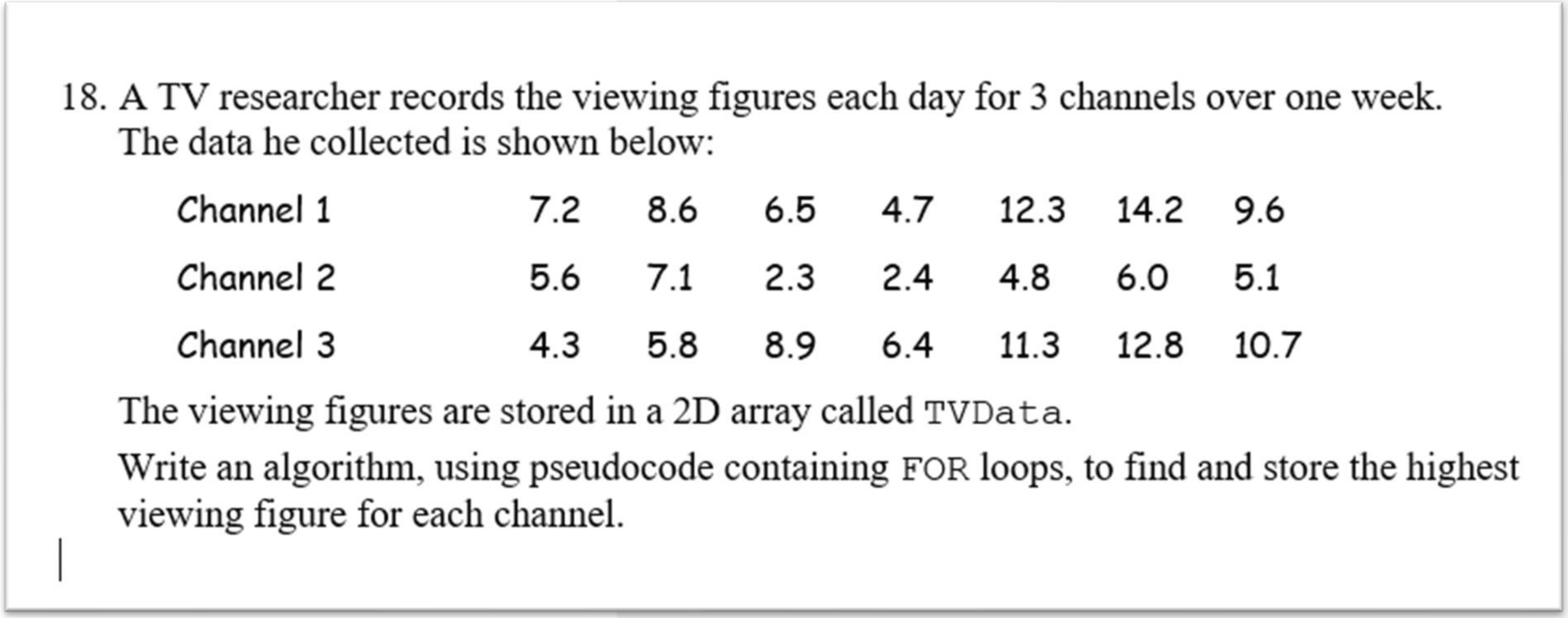
17. A store manager is thinking about prices of the 250 items in her store.
Last year's prices are recorded in a 1D array called OldPrices.
To work out each item's new price, the manager needs to multiply its price from last year by the number 1.05
The variable ItemNo stores the number of items for sale in her store.
Write an algorithm, using pseudocode containing a FOR loop, to calculate the new price for each item, storing the prices in an array.



```
17. DECLARE Count : INTEGER
    CONSTANT Multiplier ← 1.05
    CONSTANT ItemNo ← 250
    DECLARE NewPrices : ARRAY [1:ItemNo] OF REAL
    FOR Count ← 1 TO ItemNo
      NewPrices[Count] ← OldPrices[Count] * Multiplier
    NEXT Count
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



18. A TV researcher records the viewing figures each day for 3 channels over one week. The data he collected is shown below:

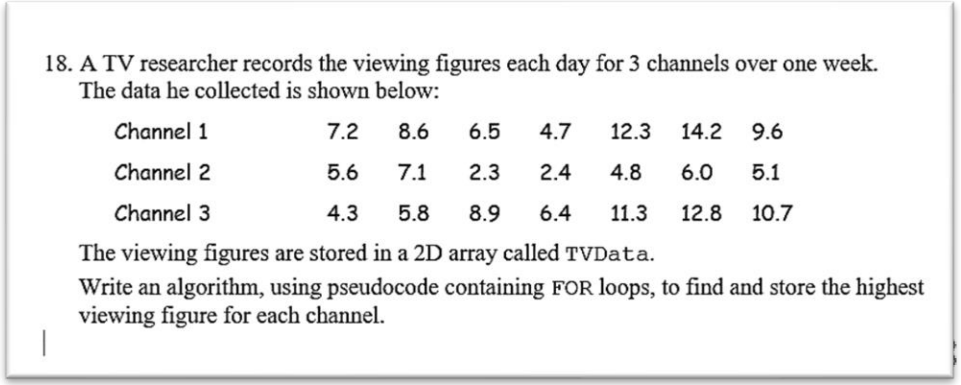
Channel 1	7.2	8.6	6.5	4.7	12.3	14.2	9.6
Channel 2	5.6	7.1	2.3	2.4	4.8	6.0	5.1
Channel 3	4.3	5.8	8.9	6.4	11.3	12.8	10.7

The viewing figures are stored in a 2D array called `TVData`.

Write an algorithm, using pseudocode containing `FOR` loops, to find and store the highest viewing figure for each channel.

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

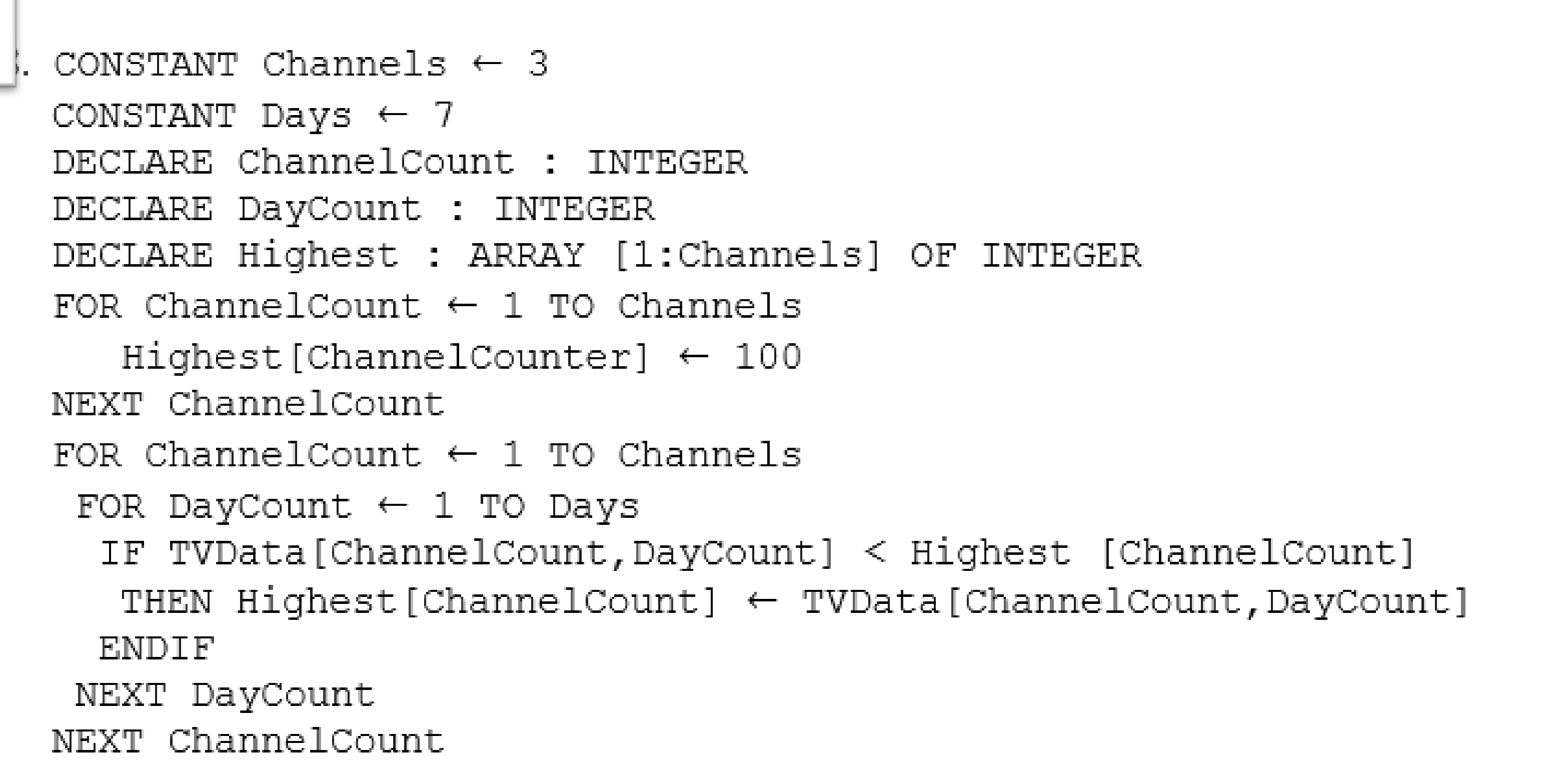


18. A TV researcher records the viewing figures each day for 3 channels over one week. The data he collected is shown below:

Channel 1	7.2	8.6	6.5	4.7	12.3	14.2	9.6
Channel 2	5.6	7.1	2.3	2.4	4.8	6.0	5.1
Channel 3	4.3	5.8	8.9	6.4	11.3	12.8	10.7

The viewing figures are stored in a 2D array called `TVData`.

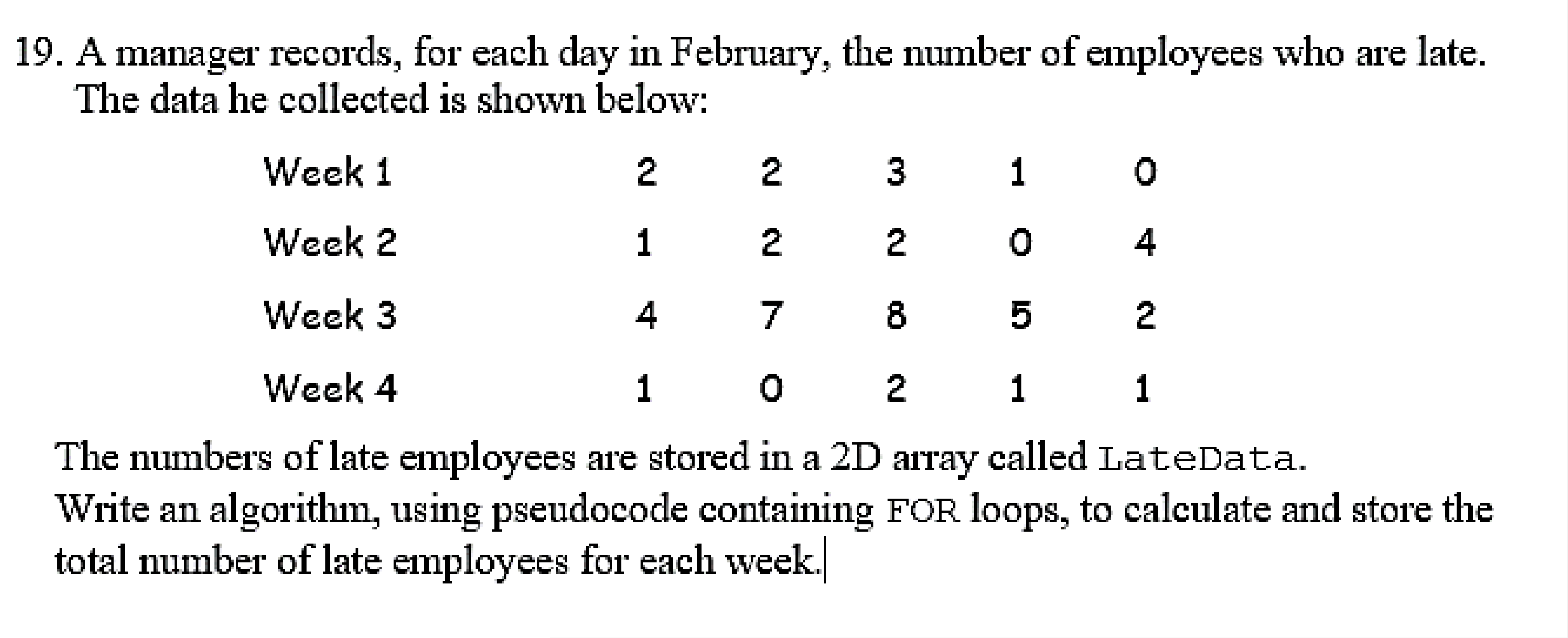
Write an algorithm, using pseudocode containing `FOR` loops, to find and store the highest viewing figure for each channel.



```
. CONSTANT Channels ← 3
CONSTANT Days ← 7
DECLARE ChannelCount : INTEGER
DECLARE DayCount : INTEGER
DECLARE Highest : ARRAY [1:Channels] OF INTEGER
FOR ChannelCount ← 1 TO Channels
   Highest[ChannelCounter] ← 100
NEXT ChannelCount
FOR ChannelCount ← 1 TO Channels
 FOR DayCount ← 1 TO Days
  IF TVData[ChannelCount,DayCount] < Highest [ChannelCount]
   THEN Highest[ChannelCount] ← TVData[ChannelCount,DayCount]
  ENDIF
 NEXT DayCount
NEXT ChannelCount
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



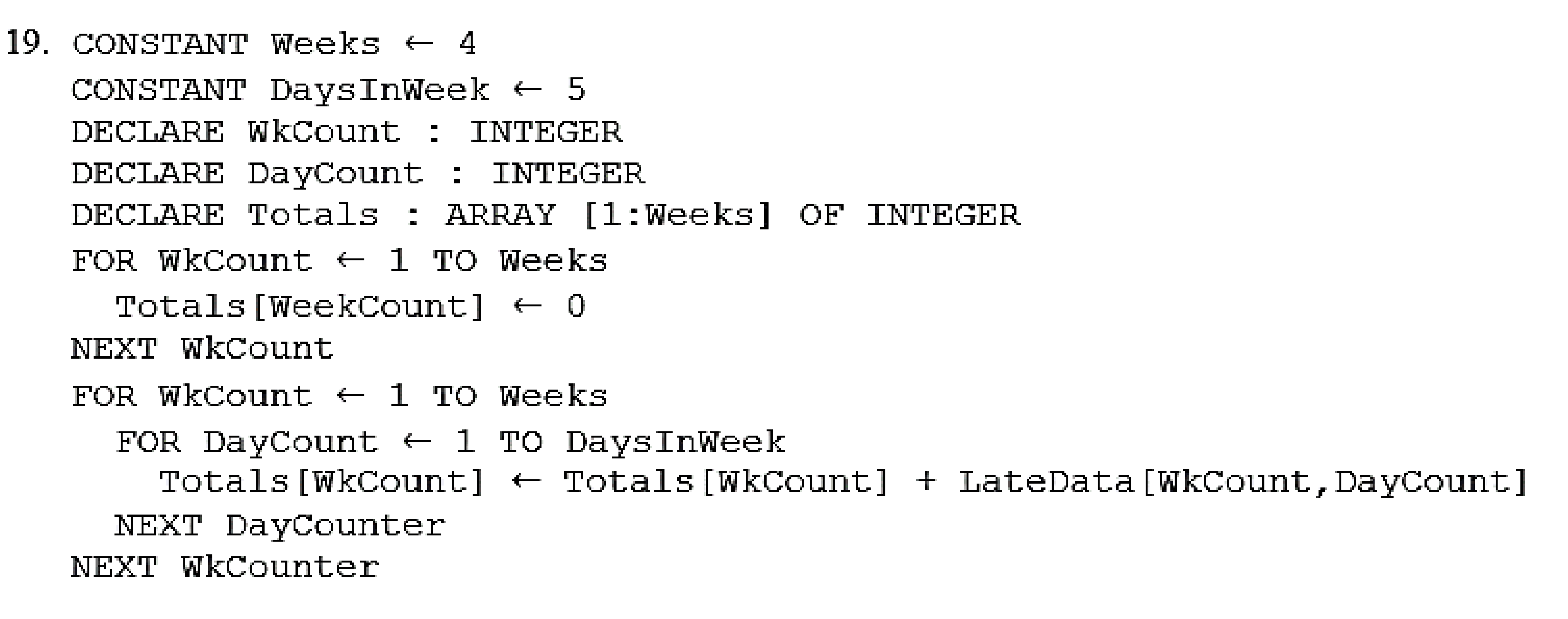
19. A manager records, for each day in February, the number of employees who are late. The data he collected is shown below:

Week 1	2	2	3	1	0
Week 2	1	2	2	0	4
Week 3	4	7	8	5	2
Week 4	1	0	2	1	1

The numbers of late employees are stored in a 2D array called `LateData`.
Write an algorithm, using pseudocode containing `FOR` loops, to calculate and store the total number of late employees for each week.

For loops worded problems

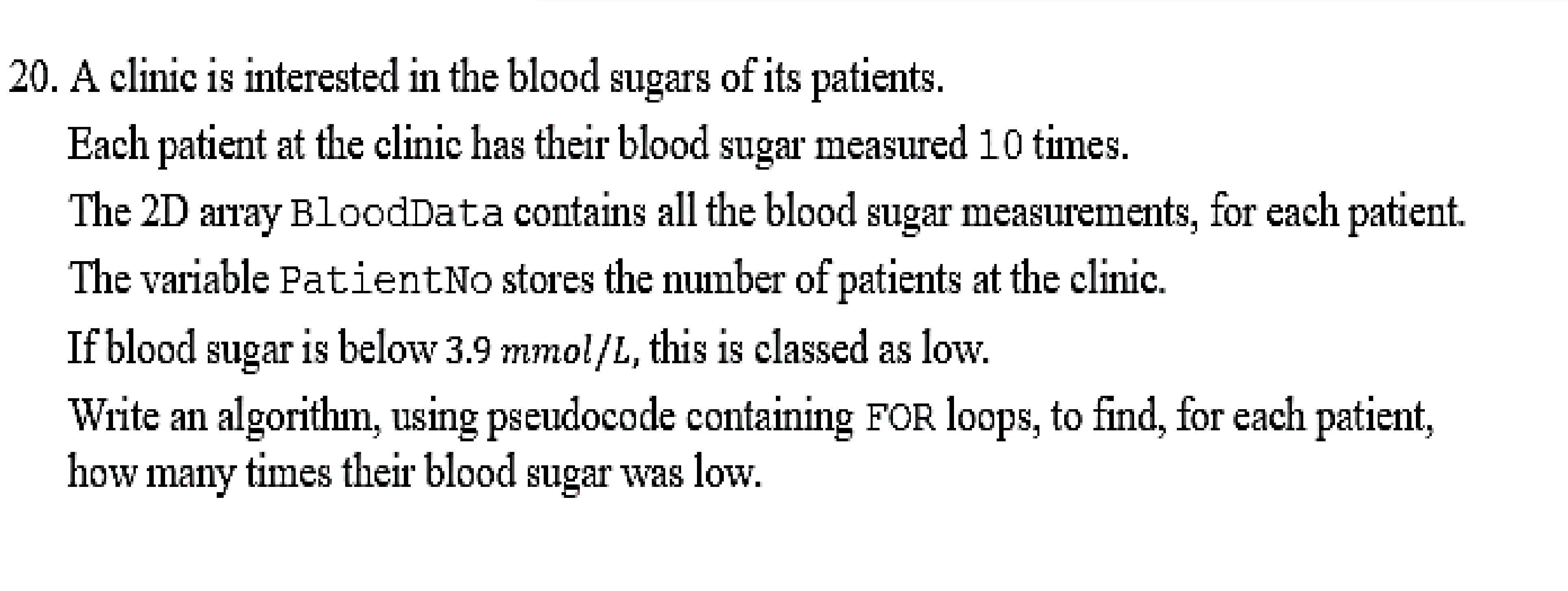
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



19.
```
CONSTANT Weeks ← 4
CONSTANT DaysInWeek ← 5
DECLARE WkCount : INTEGER
DECLARE DayCount : INTEGER
DECLARE Totals : ARRAY [1:Weeks] OF INTEGER
FOR WkCount ← 1 TO Weeks
  Totals[WeekCount] ← 0
NEXT WkCount
FOR WkCount ← 1 TO Weeks
  FOR DayCount ← 1 TO DaysInWeek
    Totals[WkCount] ← Totals[WkCount] + LateData[WkCount,DayCount]
  NEXT DayCounter
NEXT WkCounter
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



20. A clinic is interested in the blood sugars of its patients.

    Each patient at the clinic has their blood sugar measured `10` times.

    The 2D array `BloodData` contains all the blood sugar measurements, for each patient.

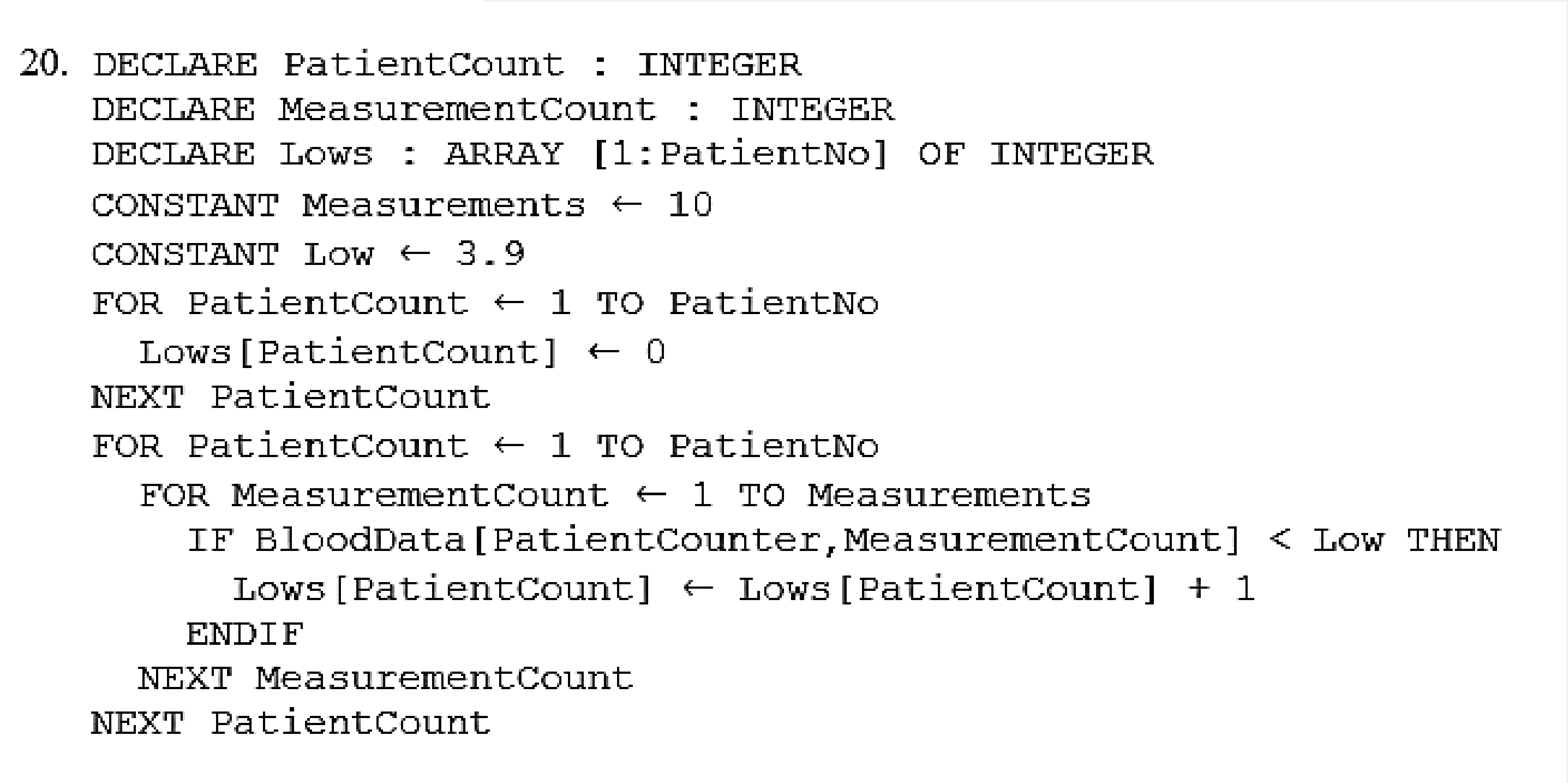
    The variable `PatientNo` stores the number of patients at the clinic.

    If blood sugar is below $3.9\ mmol/L$, this is classed as low.

    Write an algorithm, using pseudocode containing `FOR` loops, to find, for each patient, how many times their blood sugar was low.

For loops worded problems

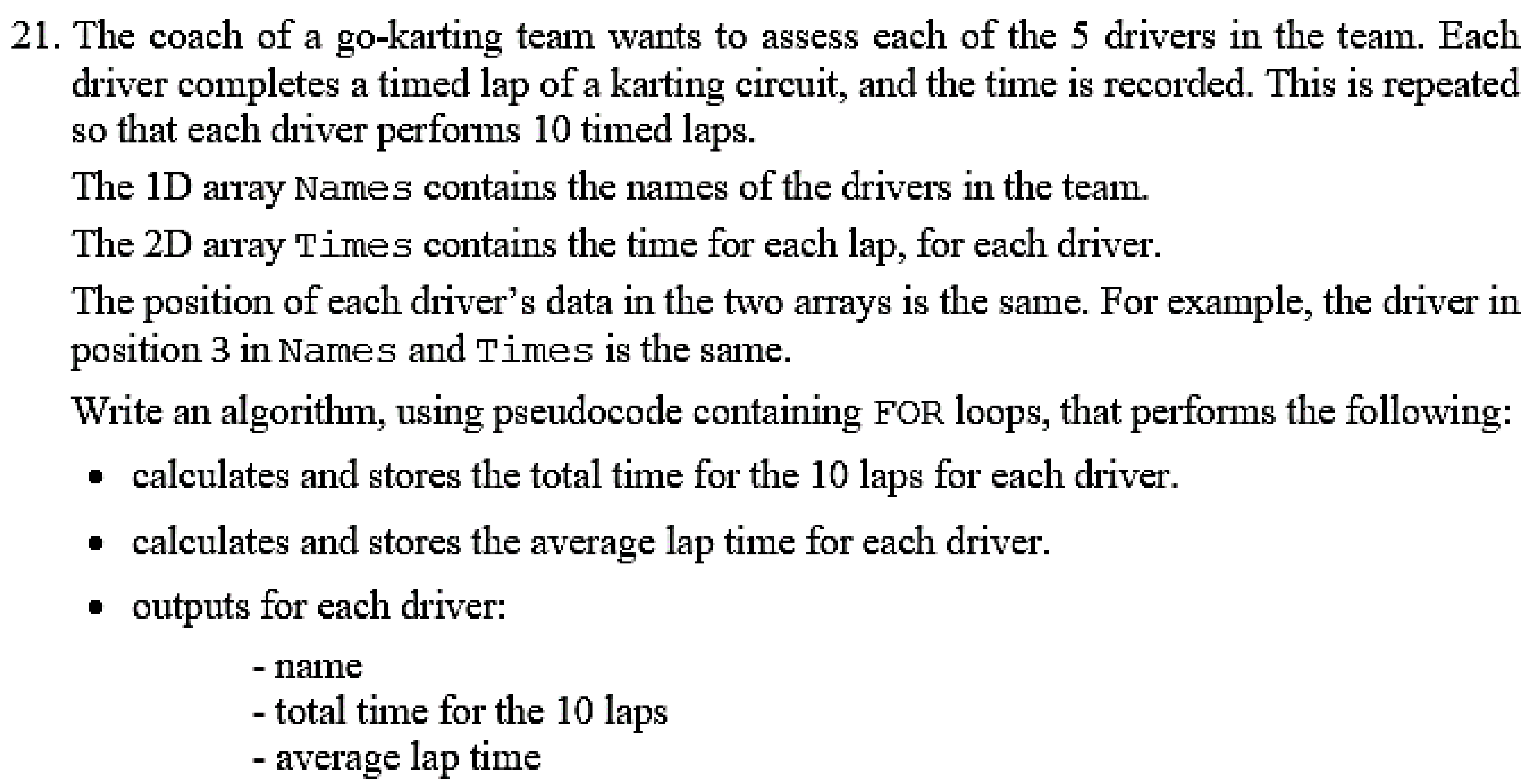
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



```
20. DECLARE PatientCount : INTEGER
    DECLARE MeasurementCount : INTEGER
    DECLARE Lows : ARRAY [1:PatientNo] OF INTEGER
    CONSTANT Measurements ← 10
    CONSTANT Low ← 3.9
    FOR PatientCount ← 1 TO PatientNo
      Lows[PatientCount] ← 0
    NEXT PatientCount
    FOR PatientCount ← 1 TO PatientNo
      FOR MeasurementCount ← 1 TO Measurements
        IF BloodData[PatientCounter,MeasurementCount] < Low THEN
          Lows[PatientCount] ← Lows[PatientCount] + 1
        ENDIF
      NEXT MeasurementCount
    NEXT PatientCount
```

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



21. The coach of a go-karting team wants to assess each of the 5 drivers in the team. Each driver completes a timed lap of a karting circuit, and the time is recorded. This is repeated so that each driver performs 10 timed laps.

    The 1D array `Names` contains the names of the drivers in the team.

    The 2D array `Times` contains the time for each lap, for each driver.

    The position of each driver's data in the two arrays is the same. For example, the driver in position 3 in `Names` and `Times` is the same.

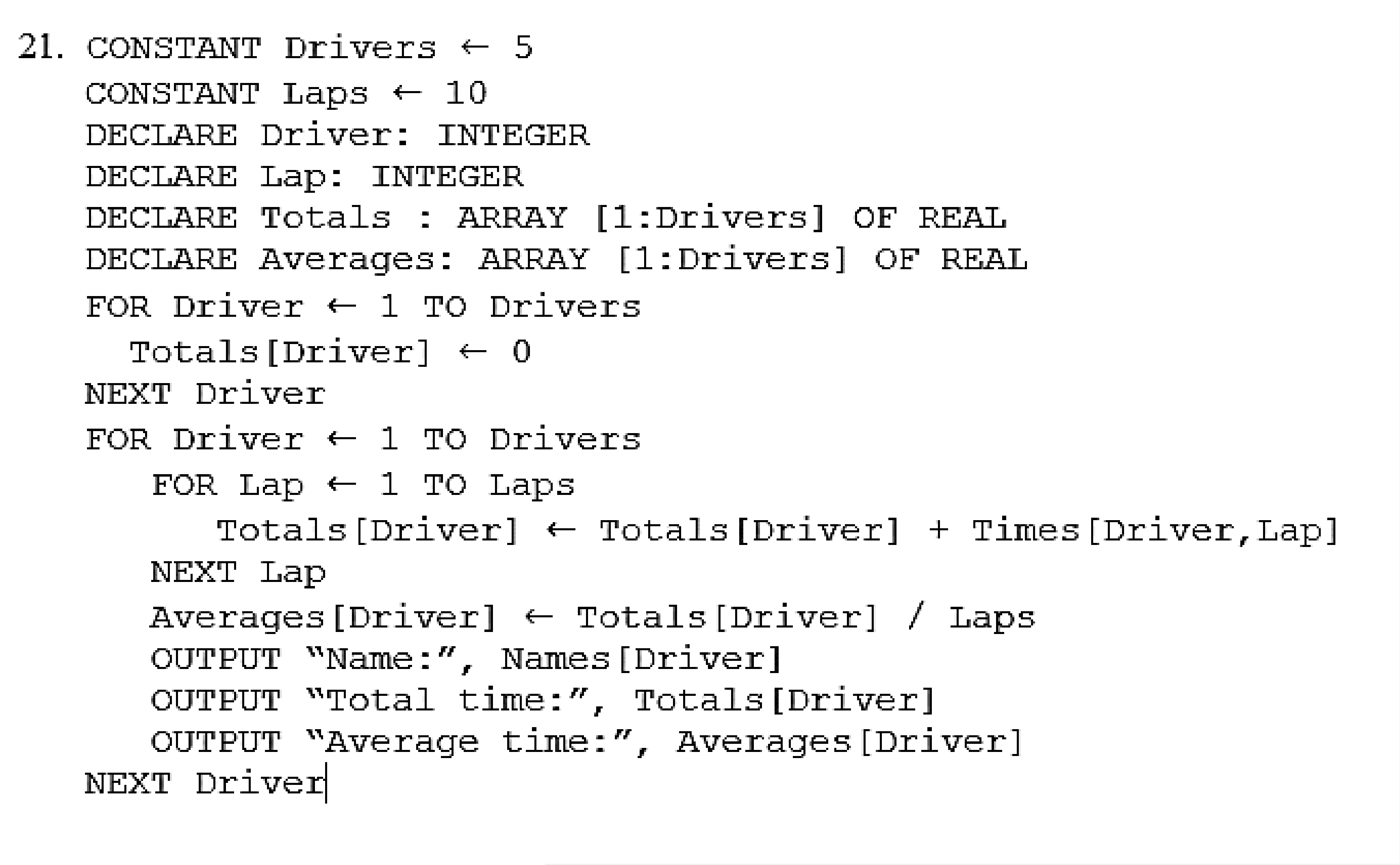
    Write an algorithm, using pseudocode containing `FOR` loops, that performs the following:

    - calculates and stores the total time for the 10 laps for each driver.
    - calculates and stores the average lap time for each driver.
    - outputs for each driver:

      - name
      - total time for the 10 laps
      - average lap time

For loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.



```
21. CONSTANT Drivers ← 5
    CONSTANT Laps ← 10
    DECLARE Driver: INTEGER
    DECLARE Lap: INTEGER
    DECLARE Totals : ARRAY [1:Drivers] OF REAL
    DECLARE Averages: ARRAY [1:Drivers] OF REAL
    FOR Driver ← 1 TO Drivers
      Totals[Driver] ← 0
    NEXT Driver
    FOR Driver ← 1 TO Drivers
       FOR Lap ← 1 TO Laps
          Totals[Driver] ← Totals[Driver] + Times[Driver,Lap]
       NEXT Lap
       Averages[Driver] ← Totals[Driver] / Laps
       OUTPUT "Name:", Names[Driver]
       OUTPUT "Total time:", Totals[Driver]
       OUTPUT "Average time:", Averages[Driver]
    NEXT Driver
```

USING

THE WHILE LOOP

For loops

Example 1

```
x = 1
while x <= 10:
   print(x)
   x = x + 1
```

Example 2

```
Sum = 0
while Sum < 100:
   x = int(input())
   Sum = Sum + x
print(Sum)
```

Example 3

```
Sum = 0
x = input()
while x != 0:
   Sum = Sum + x
   x = int(input())
print(Sum)
```

For loops

Example 1

```
x = 1
while x <= 10:
    print(x)
    x = x + 1
```

Example 2

```
Sum = 0
while Sum < 100:
    x = int(input())
    Sum = Sum + x
print(Sum)
```

Example 3

```
Sum = 0
x = input()
while x != 0:
    Sum = Sum + x
    x = int(input())
print(Sum)
```

Example 1

```
X ← 1
WHILE X <= 10 DO
    PRINT X
    X ← X + 1
ENDWHILE
```

For loops

Example 1

```
x = 1
while x <= 10:
    print(x)
    x = x + 1
```

Example 2

```
Sum = 0
while Sum < 100:
    x = int(input())
    Sum = Sum + x
print(Sum)
```

Example 3

```
Sum = 0
x = input()
while x != 0:
    Sum = Sum + x
    x = int(input())
print(Sum)
```

Example 1

```
X ← 1
WHILE X <= 10 DO
    PRINT X
    X ← X + 1
ENDWHILE
```

Example 2

```
Sum ← 0
WHILE Sum < 100 DO
    INPUT X
    Sum ← Sum + X
ENDWHILE
PRINT Sum
```

For loops

Example 1

```
x = 1
while x <= 10:
    print(x)
    x = x + 1
```

Example 2

```
Sum = 0
while Sum < 100:
    x = int(input())
    Sum = Sum + x
print(Sum)
```

Example 3

```
Sum = 0
x = input()
while x != 0:
    Sum = Sum + x
    x = int(input())
print(Sum)
```

Example 1

```
X ← 1
WHILE X <= 10 DO
    PRINT X
    X ← X + 1
ENDWHILE
```

Example 2

```
Sum ← 0
WHILE Sum < 100 DO
    INPUT X
    Sum ← Sum + X
ENDWHILE
PRINT Sum
```

Example 3

```
Sum ← 0
INPUT X
WHILE X <> 0 DO
    Sum ← Sum + X
    INPUT X
ENDWHILE
PRINT Sum
```

WHILE loops

```
x = 1
while x <= 5:
    print(x)
    x = x + 1


Count = 0
while Count < 3:
    print(Count)
    Count = Count + 1


Number = int(input())
Power = 0
while Power <= 10:
    print(Number ** Power)
    Power = Power + 1


x = int(input())
Multiple = 0
while Multiple < 1000:
    Multiple = Multiple + x
    print(Multiple)
```

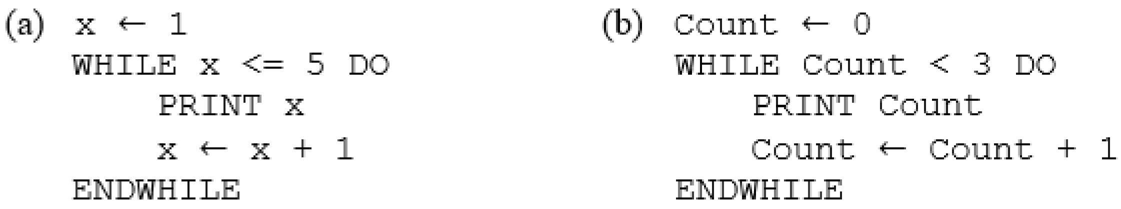
WHILE loops

```
x = 1
while x <= 5:
    print(x)
    x = x + 1


Count = 0
while Count < 3:
    print(Count)
    Count = Count + 1


Number = int(input())
Power = 0
while Power <= 10:
    print(Number ** Power)
    Power = Power + 1


x = int(input())
Multiple = 0
while Multiple < 1000:
    Multiple = Multiple + x
    print(Multiple)
```



(a)
```
x ← 1
WHILE x <= 5 DO
    PRINT x
    x ← x + 1
ENDWHILE
```

(b)
```
Count ← 0
WHILE Count < 3 DO
    PRINT Count
    Count ← Count + 1
ENDWHILE
```

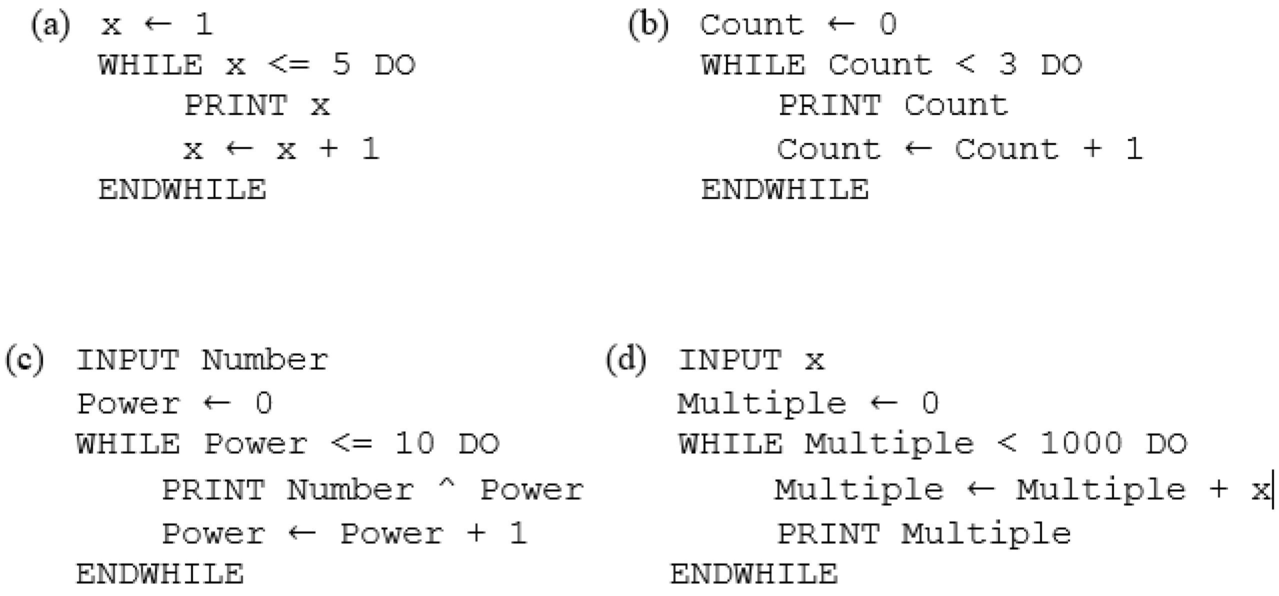
WHILE loops

```
x = 1
while x <= 5:
    print(x)
    x = x + 1


Count = 0
while Count < 3:
    print(Count)
    Count = Count + 1


Number = int(input())
Power = 0
while Power <= 10:
    print(Number ** Power)
    Power = Power + 1


x = int(input())
Multiple = 0
while Multiple < 1000:
    Multiple = Multiple + x
    print(Multiple)
```



(a)
```
x ← 1
WHILE x <= 5 DO
    PRINT x
    x ← x + 1
ENDWHILE
```

(b)
```
Count ← 0
WHILE Count < 3 DO
    PRINT Count
    Count ← Count + 1
ENDWHILE
```

(c)
```
INPUT Number
Power ← 0
WHILE Power <= 10 DO
    PRINT Number ^ Power
    Power ← Power + 1
ENDWHILE
```

(d)
```
INPUT x
Multiple ← 0
WHILE Multiple < 1000 DO
    Multiple ← Multiple + x
    PRINT Multiple
ENDWHILE
```

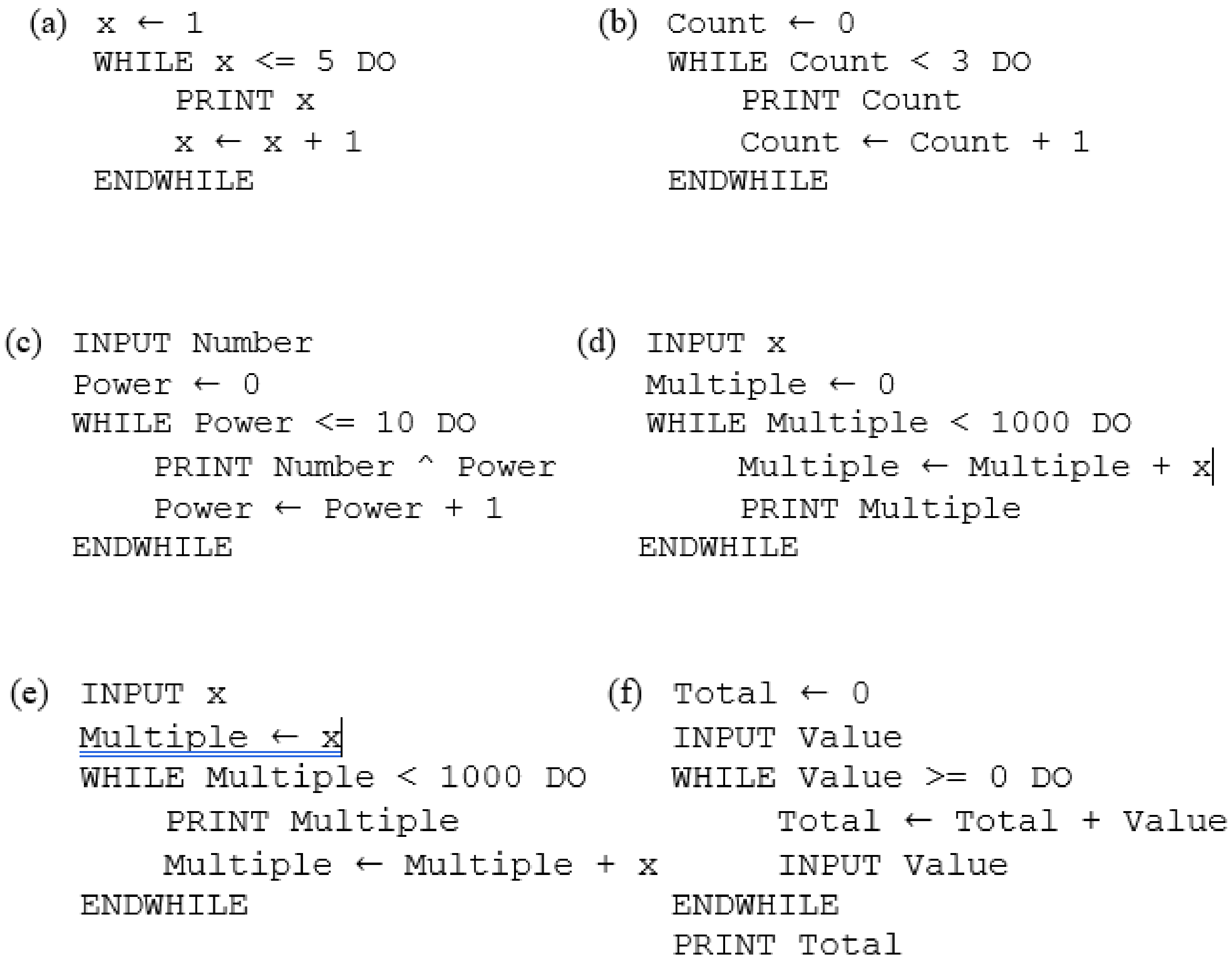
WHILE loops

```
x = 1
while x <= 5:
    print(x)
    x = x + 1


Count = 0
while Count < 3:
    print(Count)
    Count = Count + 1


Number = int(input())
Power = 0
while Power <= 10:
    print(Number ** Power)
    Power = Power + 1


x = int(input())
Multiple = 0
while Multiple < 1000:
    Multiple = Multiple + x
    print(Multiple)
```



(a)

```
x ← 1
WHILE x <= 5 DO
    PRINT x
    x ← x + 1
ENDWHILE
```

(b)

```
Count ← 0
WHILE Count < 3 DO
    PRINT Count
    Count ← Count + 1
ENDWHILE
```

(c)

```
INPUT Number
Power ← 0
WHILE Power <= 10 DO
    PRINT Number ^ Power
    Power ← Power + 1
ENDWHILE
```

(d)

```
INPUT x
Multiple ← 0
WHILE Multiple < 1000 DO
    Multiple ← Multiple + x
    PRINT Multiple
ENDWHILE
```

(e)

```
INPUT x
Multiple ← x
WHILE Multiple < 1000 DO
    PRINT Multiple
    Multiple ← Multiple + x
ENDWHILE
```

(f)

```
Total ← 0
INPUT Value
WHILE Value >= 0 DO
    Total ← Total + Value
    INPUT Value
ENDWHILE
PRINT Total
```

WHILE loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a WHILE loop, to print "Hello" five times.

2. Write an algorithm, using pseudocode with a WHILE loop, to print all the numbers 1 to 20.

3. Write an algorithm, using pseudocode containing a `WHILE` loop, to do the following 5 times:

   - ask the user to enter a positive whole number
   - output the value that is the remainder when the number entered is divided by 7

WHILE loops worded problems

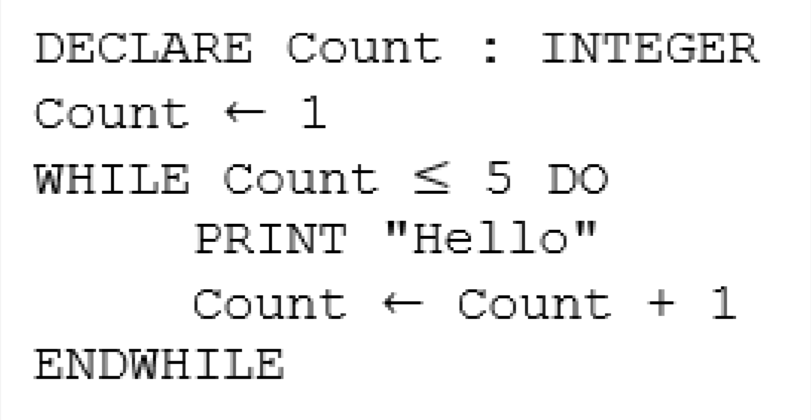
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a WHILE loop, to print "Hello" five times.

2. Write an algorithm, using pseudocode with a WHILE loop, to print all the numbers 1 to 20.

3. Write an algorithm, using pseudocode containing a `WHILE` loop, to do the following 5 times:

   - ask the user to enter a positive whole number
   - output the value that is the remainder when the number entered is divided by 7



```
DECLARE Count : INTEGER
Count ← 1
WHILE Count ≤ 5 DO
      PRINT "Hello"
      Count ← Count + 1
ENDWHILE
```

WHILE loops worded problems

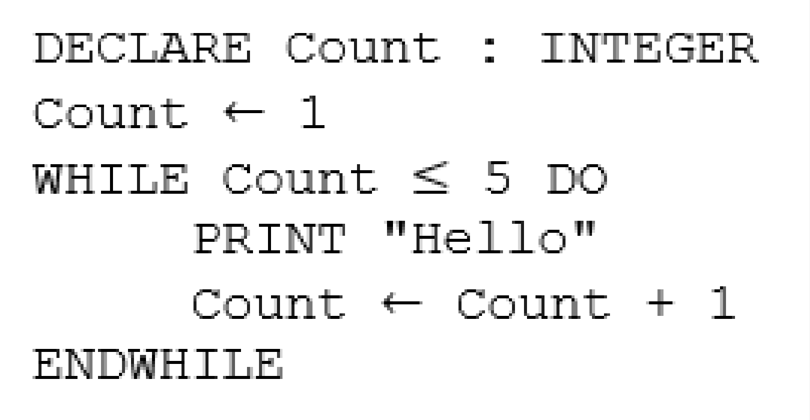
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a WHILE loop, to print "Hello" five times.

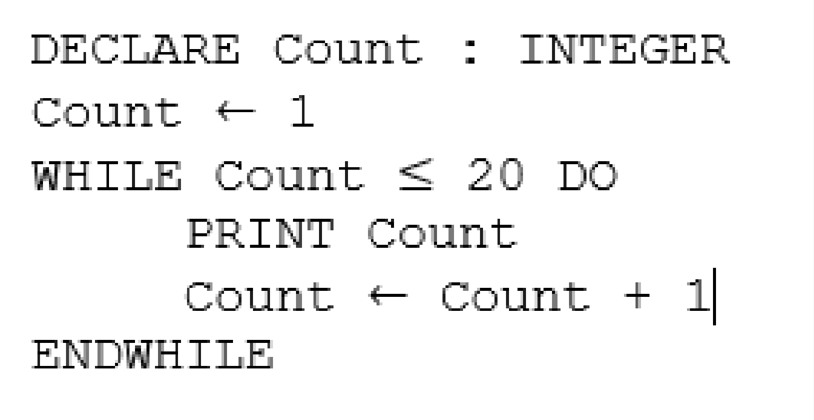
2. Write an algorithm, using pseudocode with a WHILE loop, to print all the numbers 1 to 20.

3. Write an algorithm, using pseudocode containing a `WHILE` loop, to do the following 5 times:

    - ask the user to enter a positive whole number
    - output the value that is the remainder when the number entered is divided by 7



```
DECLARE Count : INTEGER
Count ← 1
WHILE Count ≤ 5 DO
     PRINT "Hello"
     Count ← Count + 1
ENDWHILE
```

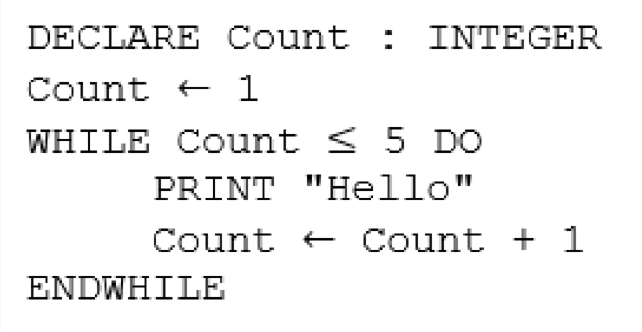


```
DECLARE Count : INTEGER
Count ← 1
WHILE Count ≤ 20 DO
     PRINT Count
     Count ← Count + 1
ENDWHILE
```

WHILE loops worded problems

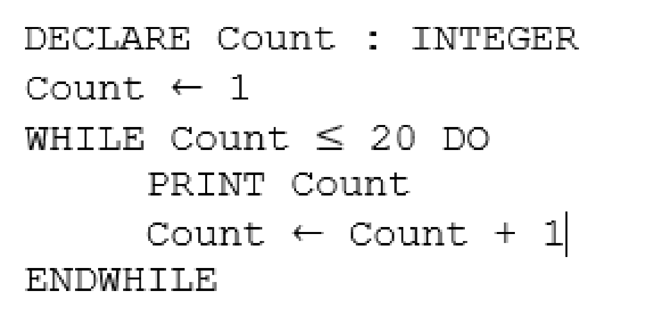
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a WHILE loop, to print "Hello" five times.



```
DECLARE Count : INTEGER
Count ← 1
WHILE Count ≤ 5 DO
     PRINT "Hello"
     Count ← Count + 1
ENDWHILE
```

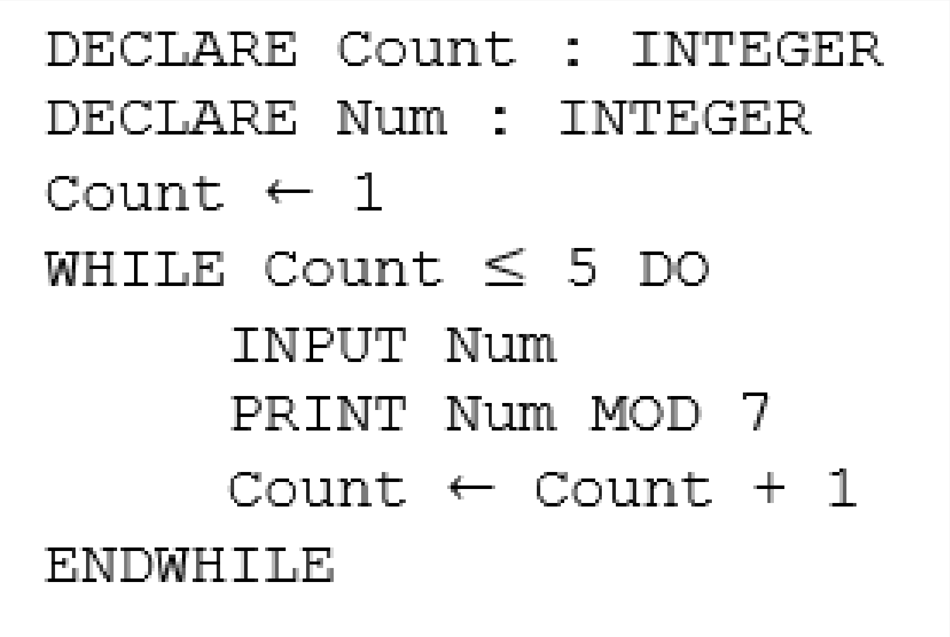
2. Write an algorithm, using pseudocode with a WHILE loop, to print all the numbers 1 to 20.



```
DECLARE Count : INTEGER
Count ← 1
WHILE Count ≤ 20 DO
     PRINT Count
     Count ← Count + 1
ENDWHILE
```

3. Write an algorithm, using pseudocode containing a `WHILE` loop, to do the following 5 times:

   - ask the user to enter a positive whole number
   - output the value that is the remainder when the number entered is divided by 7



```
DECLARE Count : INTEGER
DECLARE Num : INTEGER
Count ← 1
WHILE Count ≤ 5 DO
     INPUT Num
     PRINT Num MOD 7
     Count ← Count + 1
ENDWHILE
```

WHILE loops worded problems

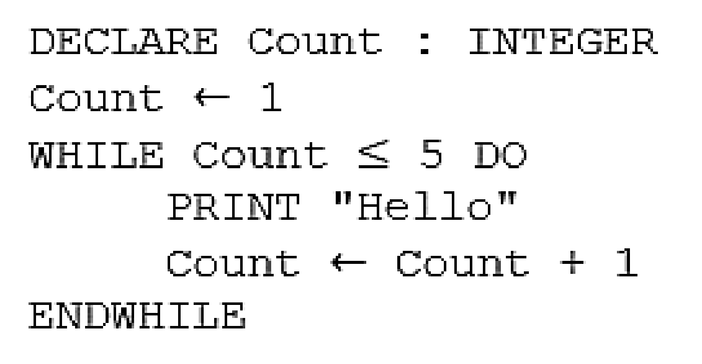
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode containin a `WHILE` loop, to:

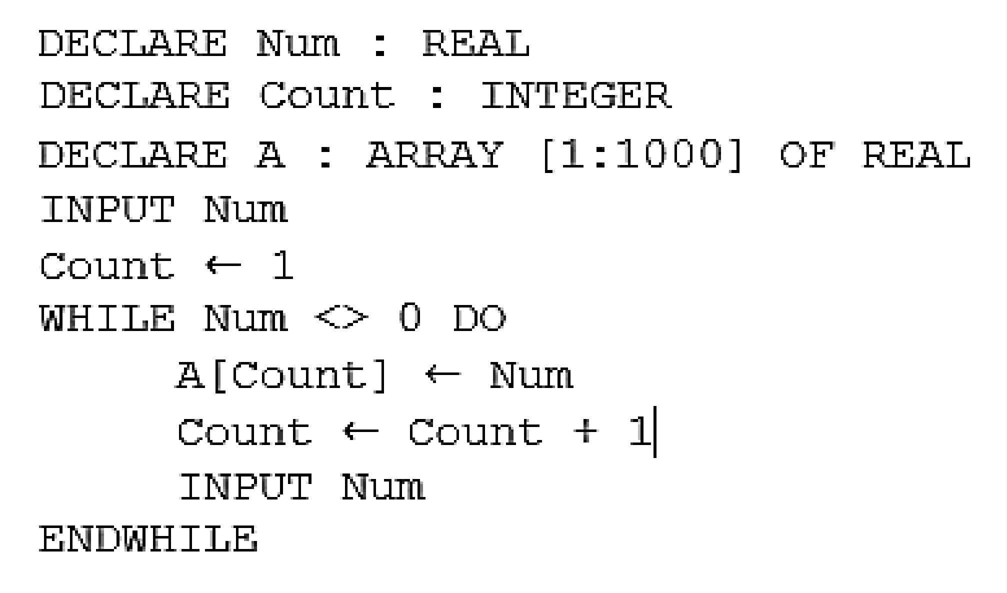
   - ask the user to enter heights of people, i metres
   - stop asking for heights when the user enter a value that is not in $0 < height < 2$
   - finds and outputs the average of the height entered

5. Write an algorithm, using pseudocode containin a `WHILE` loop, to:

   - ask the user to enter numbers
   - stop asking for numbers when the user enter zero
   - store all the entered numbers (except th zero) into an array



```
DECLARE Count : INTEGER
Count ← 1
WHILE Count ≤ 5 DO
    PRINT "Hello"
    Count ← Count + 1
ENDWHILE
```



```
DECLARE Num : REAL
DECLARE Count : INTEGER
DECLARE A : ARRAY [1:1000] OF REAL
INPUT Num
Count ← 1
WHILE Num <> 0 DO
    A[Count] ← Num
    Count ← Count + 1
    INPUT Num
ENDWHILE
```

USING

REPEAT UNTIL LOOP

REPEAT UNTIL loops

Example 1

```
X ← 1
REPEAT
   PRINT X
   X ← X + 1
UNTIL X = 11
```

Example 2

```
Sum ← 0
REPEAT
   INPUT X
   Sum ← Sum + X
UNTIL Sum > 1000
PRINT Sum
```

Example 3

```
Sum ← 0
REPEAT
   INPUT Num
   Sum ← Sum + Num
UNTIL Num = 0
PRINT Sum
```

Change to REPEAT UNTIL loops

```
FOR x ← 1 TO 5
    PRINT x
NEXT x


FOR Count ← 1 TO 100
    INPUT A[Count]
NEXT Count


INPUT Max
FOR x ← 1 TO Max
    PRINT x
NEXT x


x ← 1
WHILE x <=10 DO
    PRINT x
    x ← x + 1
ENDWHILE


Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```

Rewrite the above pseudocodes into repeat – until loop

Change to REPEAT UNTIL loops

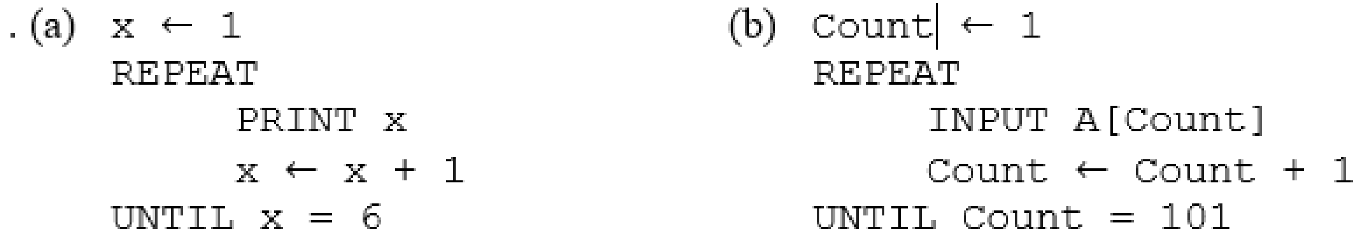
```
FOR x ← 1 TO 5
    PRINT x
NEXT x
```

```
FOR Count ← 1 TO 100
    INPUT A[Count]
NEXT Count
```

```
INPUT Max
FOR x ← 1 TO Max
    PRINT x
NEXT x
```

```
x ← 1
WHILE x <=10 DO
    PRINT x
    x ← x + 1
ENDWHILE
```

```
Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```



. (a)

```
x ← 1
REPEAT
    PRINT x
    x ← x + 1
UNTIL x = 6
```

(b)

```
Count ← 1
REPEAT
    INPUT A[Count]
    Count ← Count + 1
UNTIL Count = 101
```

Rewrite the above pseudocodes into repeat – until loop

Change to REPEAT UNTIL loops

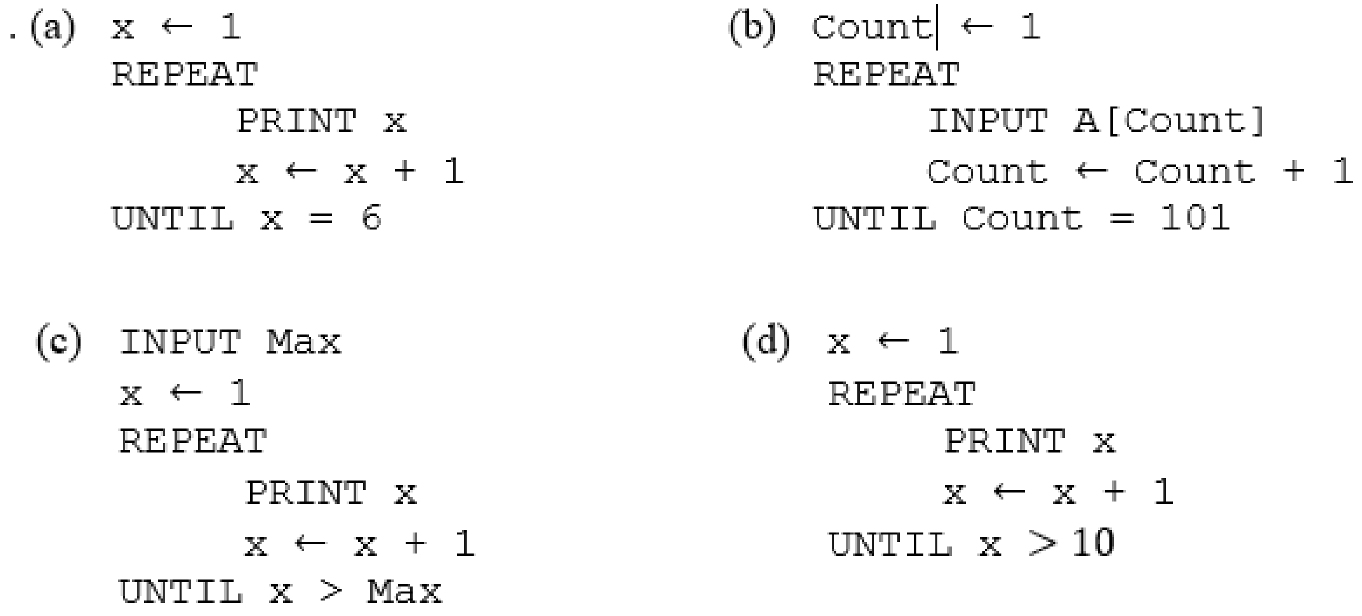
```
FOR x ← 1 TO 5
    PRINT x
NEXT x
```

```
FOR Count ← 1 TO 100
    INPUT A[Count]
NEXT Count
```

```
INPUT Max
FOR x ← 1 TO Max
    PRINT x
NEXT x
```

```
x ← 1
WHILE x <=10 DO
    PRINT x
    x ← x + 1
ENDWHILE
```

```
Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```



. (a)

```
x ← 1
REPEAT
    PRINT x
    x ← x + 1
UNTIL x = 6
```

(b)

```
Count ← 1
REPEAT
    INPUT A[Count]
    Count ← Count + 1
UNTIL Count = 101
```

(c)

```
INPUT Max
x ← 1
REPEAT
    PRINT x
    x ← x + 1
UNTIL x > Max
```

(d)

```
x ← 1
REPEAT
    PRINT x
    x ← x + 1
UNTIL x > 10
```

Rewrite the above pseudocodes into repeat – until loop

Change to REPEAT UNTIL loops

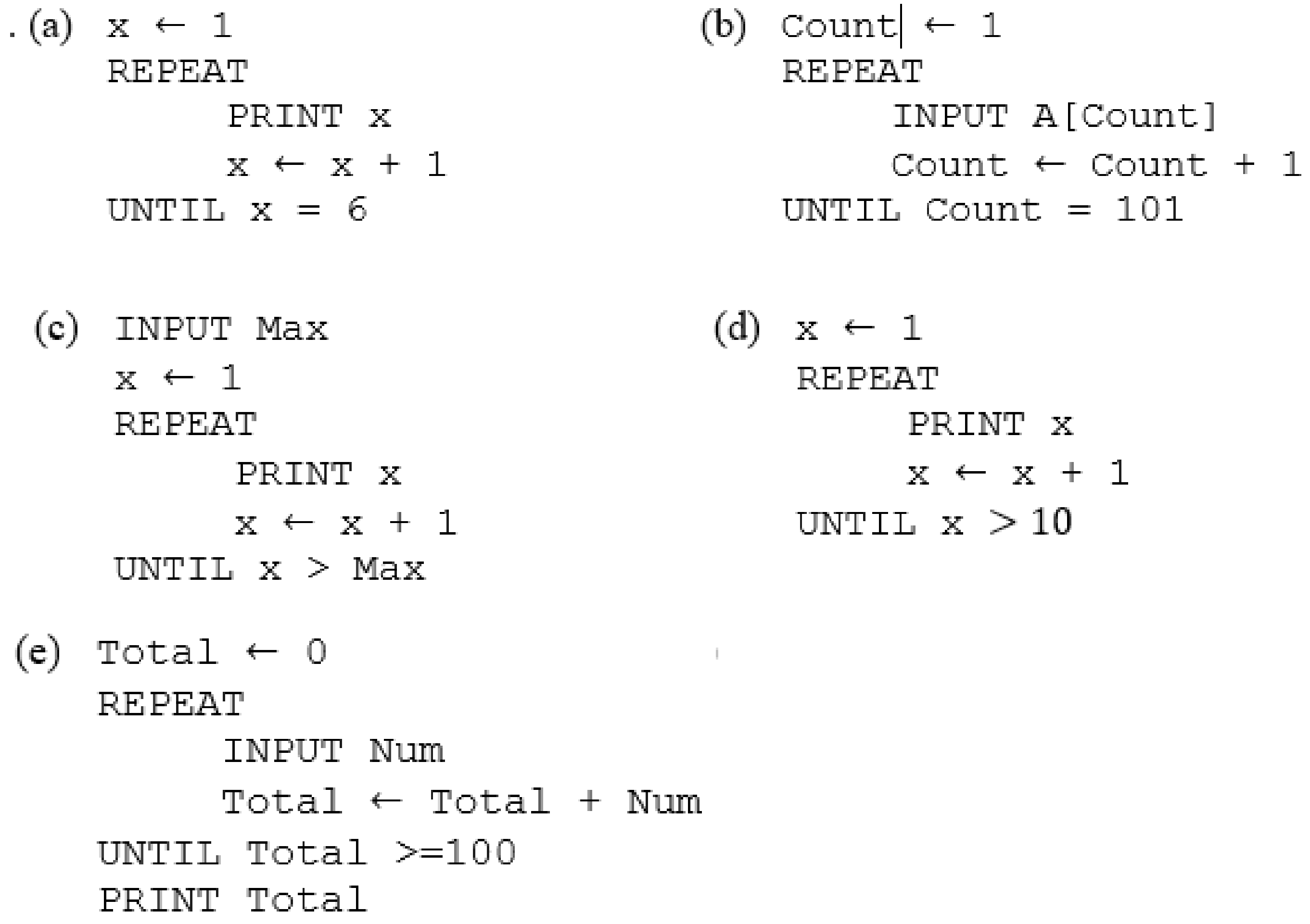
```
FOR x ← 1 TO 5
    PRINT x
NEXT x


FOR Count ← 1 TO 100
    INPUT A[Count]
NEXT Count


INPUT Max
FOR x ← 1 TO Max
    PRINT x
NEXT x


x ← 1
WHILE x <=10 DO
    PRINT x
    x ← x + 1
ENDWHILE


Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```



```
. (a)  x ← 1
       REPEAT
           PRINT x
           x ← x + 1
       UNTIL x = 6

(b)  Count ← 1
     REPEAT
         INPUT A[Count]
         Count ← Count + 1
     UNTIL Count = 101

(c)  INPUT Max
     x ← 1
     REPEAT
         PRINT x
         x ← x + 1
     UNTIL x > Max

(d)  x ← 1
     REPEAT
         PRINT x
         x ← x + 1
     UNTIL x > 10

(e)  Total ← 0
     REPEAT
         INPUT Num
         Total ← Total + Num
     UNTIL Total >=100
     PRINT Total
```

Rewrite the above pseudocodes into repeat – until loop

Change to REPEAT UNTIL loops

```
Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```

```
INPUT Num
Count ← 1
WHILE Num <> 0 DO
    A[Count] ← Num
    Count ← Count + 1
    INPUT Num
ENDWHILE
```

```
INPUT Num
WHILE Num MOD 9 = 0 DO
    PRINT "Yes, that is a multiple of 9"
    INPUT Num
ENDWHILE
PRINT "No, that is not a multiple of 9"
```

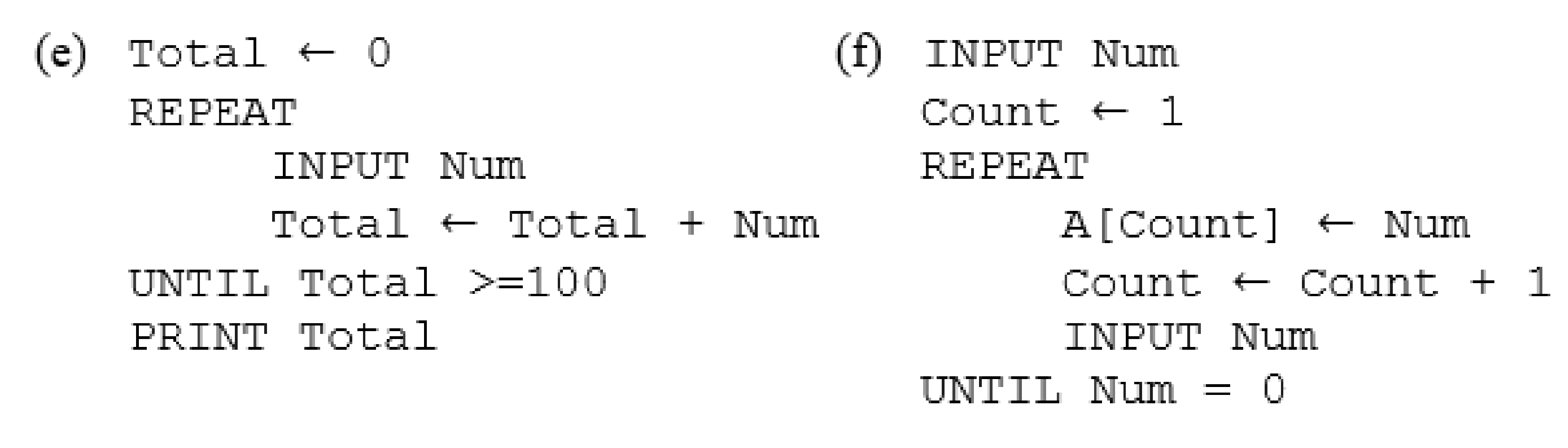
Rewrite the above pseudocodes into repeat – until loop

Change to REPEAT UNTIL loops

```
Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```

```
INPUT Num
Count ← 1
WHILE Num <> 0 DO
    A[Count] ← Num
    Count ← Count + 1
    INPUT Num
ENDWHILE
```

```
INPUT Num
WHILE Num MOD 9 = 0 DO
    PRINT "Yes, that is a multiple of 9"
    INPUT Num
ENDWHILE
PRINT "No, that is not a multiple of 9"
```



(e)

```
Total ← 0
REPEAT
    INPUT Num
    Total ← Total + Num
UNTIL Total >=100
PRINT Total
```

(f)

```
INPUT Num
Count ← 1
REPEAT
    A[Count] ← Num
    Count ← Count + 1
    INPUT Num
UNTIL Num = 0
```

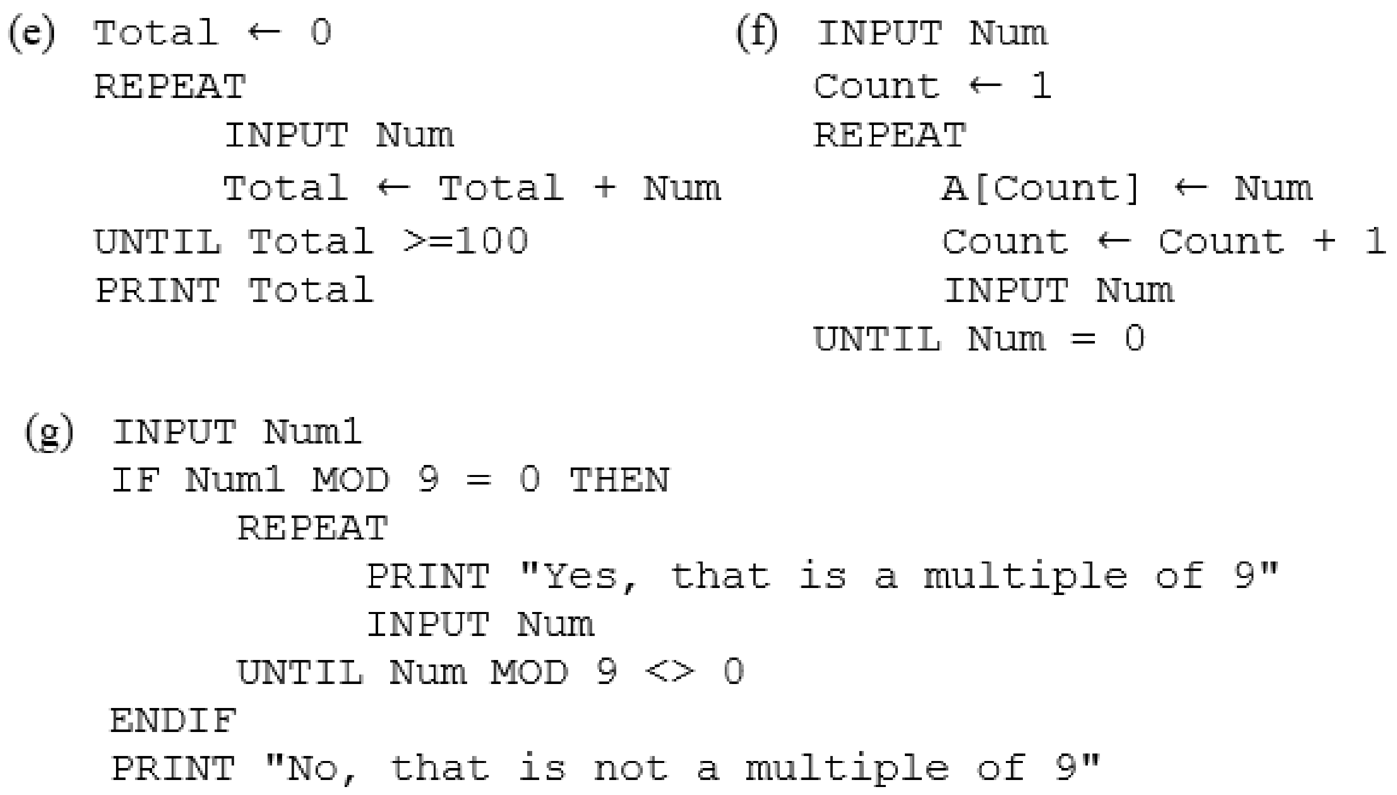
Rewrite the above pseudocodes into repeat – until loop

Change to REPEAT UNTIL loops

```
Total ← 0
WHILE Total < 100 DO
    INPUT Num
    Total ← Total + Num
ENDWHILE
PRINT Total
```

```
INPUT Num
Count ← 1
WHILE Num <> 0 DO
    A[Count] ← Num
    Count ← Count + 1
    INPUT Num
ENDWHILE
```

```
INPUT Num
WHILE Num MOD 9 = 0 DO
    PRINT "Yes, that is a multiple of 9"
    INPUT Num
ENDWHILE
PRINT "No, that is not a multiple of 9"
```



(e)

```
Total ← 0
REPEAT
    INPUT Num
    Total ← Total + Num
UNTIL Total >=100
PRINT Total
```

(f)

```
INPUT Num
Count ← 1
REPEAT
    A[Count] ← Num
    Count ← Count + 1
    INPUT Num
UNTIL Num = 0
```

(g)

```
INPUT Num1
IF Num1 MOD 9 = 0 THEN
    REPEAT
        PRINT "Yes, that is a multiple of 9"
        INPUT Num
    UNTIL Num MOD 9 <> 0
ENDIF
PRINT "No, that is not a multiple of 9"
```

Rewrite the above pseudocodes into repeat – until loop

REPEAT UNTIL loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a REPEAT loop, to print "Hello" ten times.

2. Write an algorithm, using pseudocode with a REPEAT loop, to print all the numbers 1 to 50.

3. Write an algorithm in pseudocode, containing a REPEAT loop, to do the following 5 times:

   - ask the user to enter a whole number
   - output the value that is the remainder when the number entered is divided by 3

REPEAT UNTIL loops worded problems

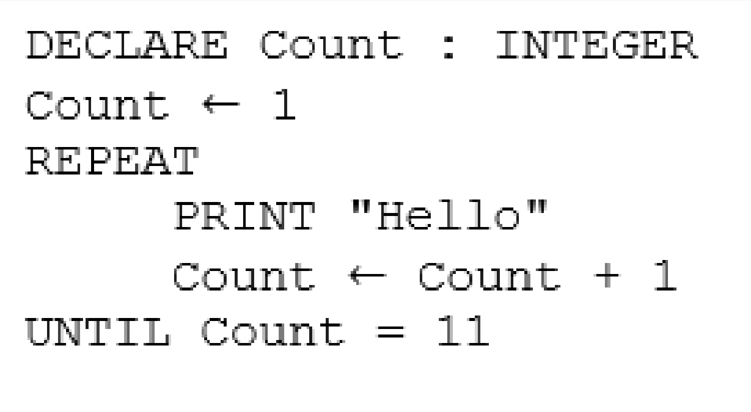
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a REPEAT loop, to print "Hello" ten times.

2. Write an algorithm, using pseudocode with a REPEAT loop, to print all the numbers 1 to 50.

3. Write an algorithm in pseudocode, containing a REPEAT loop, to do the following 5 times:

   - ask the user to enter a whole number
   - output the value that is the remainder when the number entered is divided by 3

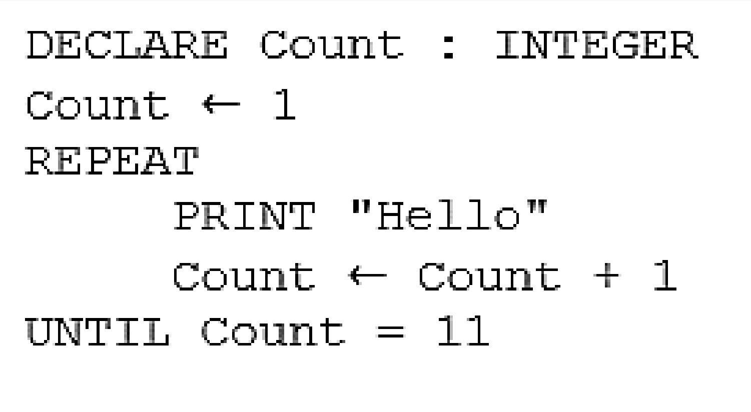


```
DECLARE Count : INTEGER
Count ← 1
REPEAT
     PRINT "Hello"
     Count ← Count + 1
UNTIL Count = 11
```

REPEAT UNTIL loops worded problems

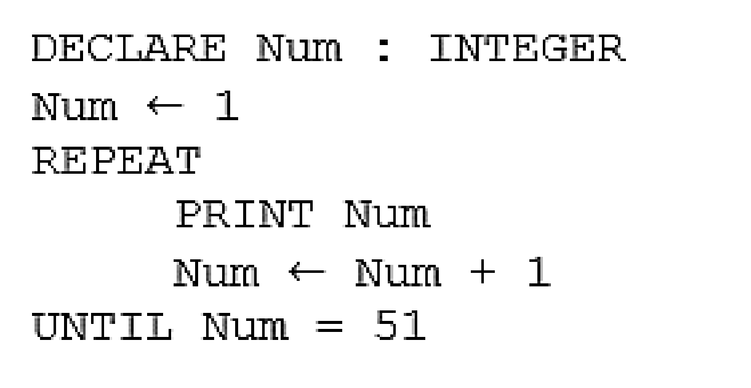
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a REPEAT loop, to print "Hello" ten times.



```
DECLARE Count : INTEGER
Count ← 1
REPEAT
     PRINT "Hello"
     Count ← Count + 1
UNTIL Count = 11
```

2. Write an algorithm, using pseudocode with a REPEAT loop, to print all the numbers 1 to 50.



```
DECLARE Num : INTEGER
Num ← 1
REPEAT
     PRINT Num
     Num ← Num + 1
UNTIL Num = 51
```

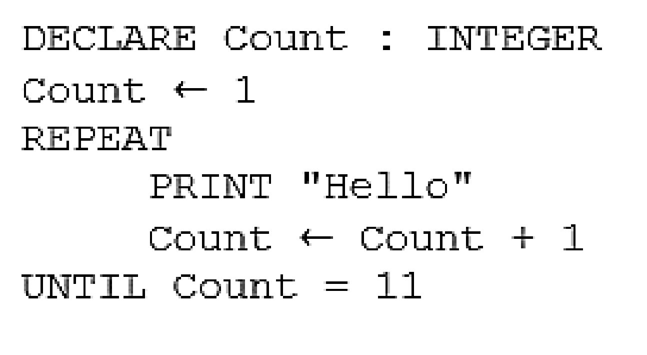
3. Write an algorithm in pseudocode, containing a REPEAT loop, to do the following 5 times:

   - ask the user to enter a whole number
   - output the value that is the remainder when the number entered is divided by 3

REPEAT UNTIL loops worded problems

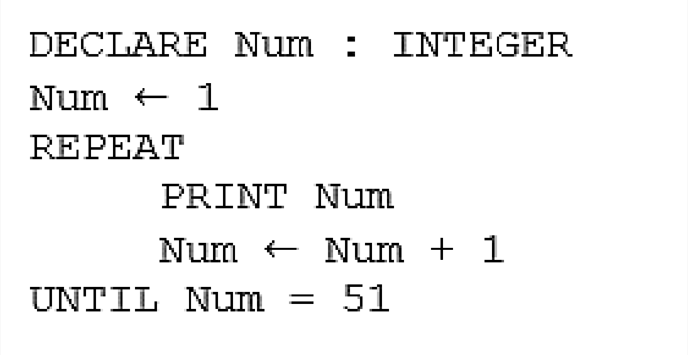
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

1. Write an algorithm, using pseudocode containing a REPEAT loop, to print "Hello" ten times.



```
DECLARE Count : INTEGER
Count ← 1
REPEAT
     PRINT "Hello"
     Count ← Count + 1
UNTIL Count = 11
```

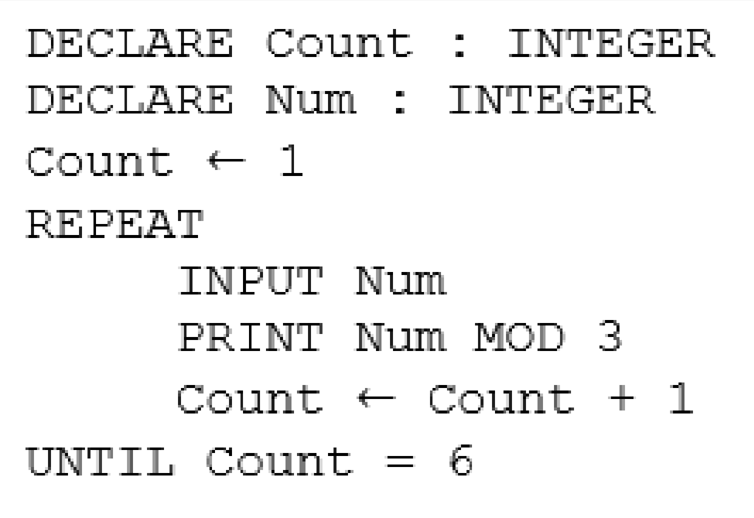
2. Write an algorithm, using pseudocode with a REPEAT loop, to print all the numbers 1 to 50.



```
DECLARE Num : INTEGER
Num ← 1
REPEAT
     PRINT Num
     Num ← Num + 1
UNTIL Num = 51
```

3. Write an algorithm in pseudocode, containing a REPEAT loop, to do the following 5 times:

   - ask the user to enter a whole number
   - output the value that is the remainder when the number entered is divided by 3



```
DECLARE Count : INTEGER
DECLARE Num : INTEGER
Count ← 1
REPEAT
     INPUT Num
     PRINT Num MOD 3
     Count ← Count + 1
UNTIL Count = 6
```

REPEAT UNTIL loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

- ask the user to enter the ages of people, in years
- stop asking for ages when the user enters an age that is not in $0 \leq age \leq 100$
- output the average of the ages entered

5. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

- ask the user to enter an odd number
- stop asking for numbers when they enter an odd number
- output the message "Valid input"

REPEAT UNTIL loops worded problems

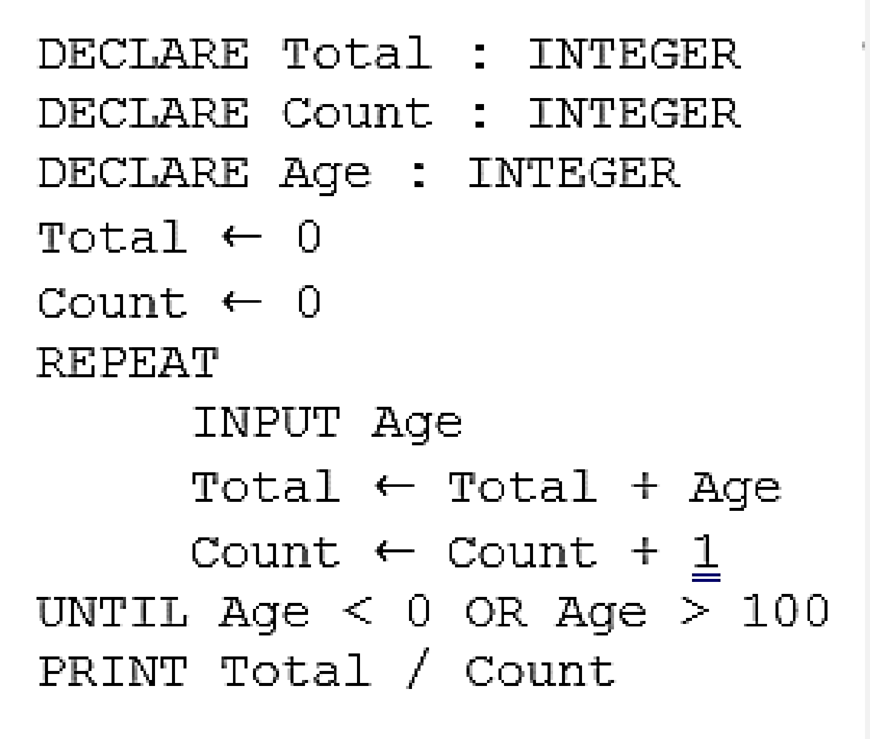
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

- ask the user to enter the ages of people, in years
- stop asking for ages when the user enters an age that is not in $0 \leq age \leq 100$
- output the average of the ages entered

5. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

- ask the user to enter an odd number
- stop asking for numbers when they enter an odd number
- output the message "Valid input"



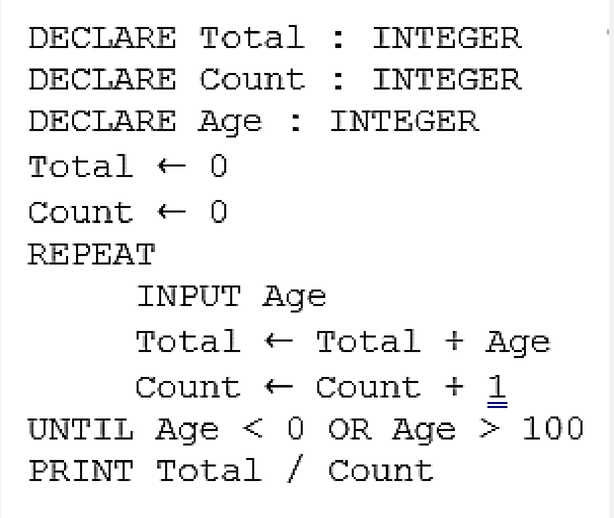
```
DECLARE Total : INTEGER
DECLARE Count : INTEGER
DECLARE Age : INTEGER
Total ← 0
Count ← 0
REPEAT
     INPUT Age
     Total ← Total + Age
     Count ← Count + 1
UNTIL Age < 0 OR Age > 100
PRINT Total / Count
```

REPEAT UNTIL loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

4. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

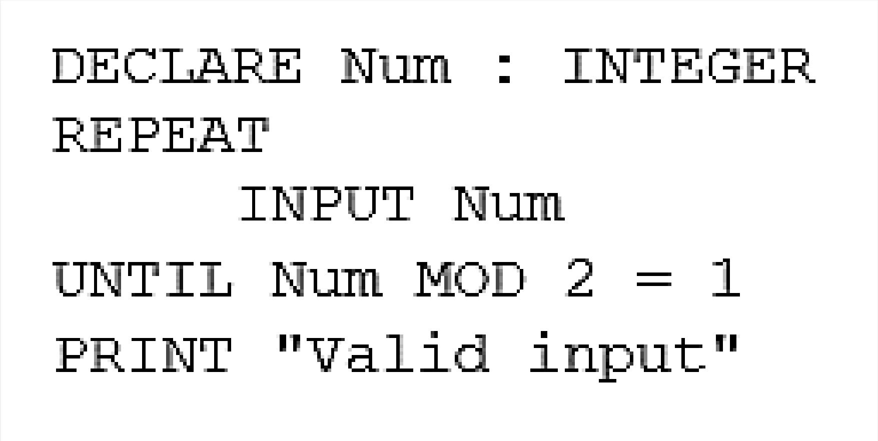
- ask the user to enter the ages of people, in years
- stop asking for ages when the user enters an age that is not in $0 \leq age \leq 100$
- output the average of the ages entered



```
DECLARE Total : INTEGER
DECLARE Count : INTEGER
DECLARE Age : INTEGER
Total ← 0
Count ← 0
REPEAT
     INPUT Age
     Total ← Total + Age
     Count ← Count + 1
UNTIL Age < 0 OR Age > 100
PRINT Total / Count
```

5. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

- ask the user to enter an odd number
- stop asking for numbers when they enter an odd number
- output the message "Valid input"



```
DECLARE Num : INTEGER
REPEAT
     INPUT Num
UNTIL Num MOD 2 = 1
PRINT "Valid input"
```

REPEAT UNTIL loops worded problems

DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

   - ask the user to enter two whole numbers that are not multiples of 7
   - stop asking for numbers when they enter two valid numbers
   - calculate and output the product of the two valid numbers entered

7. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

   - ask the user to enter numbers
   - stop asking for numbers when the user enters a negative value
   - store all the entered numbers into an array

REPEAT UNTIL loops worded problems

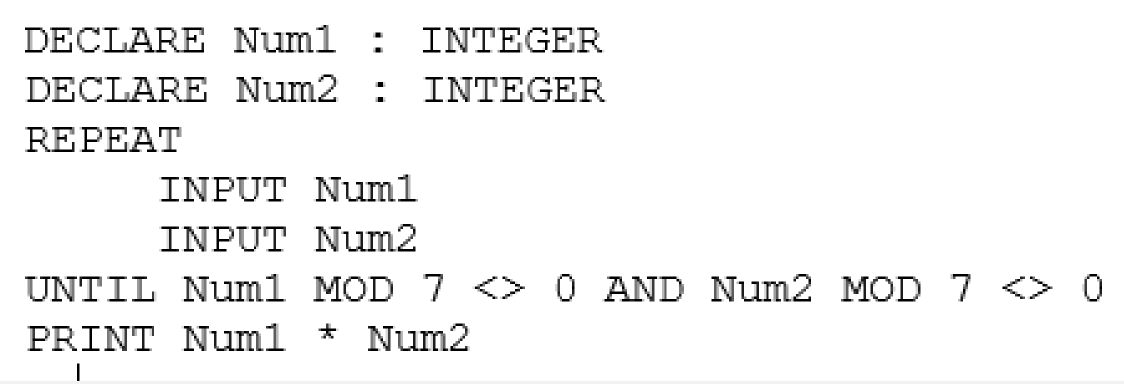
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

   - ask the user to enter two whole numbers that are not multiples of 7
   - stop asking for numbers when they enter two valid numbers
   - calculate and output the product of the two valid numbers entered

7. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

   - ask the user to enter numbers
   - stop asking for numbers when the user enters a negative value
   - store all the entered numbers into an array



```
DECLARE Num1 : INTEGER
DECLARE Num2 : INTEGER
REPEAT
      INPUT Num1
      INPUT Num2
UNTIL Num1 MOD 7 <> 0 AND Num2 MOD 7 <> 0
PRINT Num1 * Num2
```

REPEAT UNTIL loops worded problems

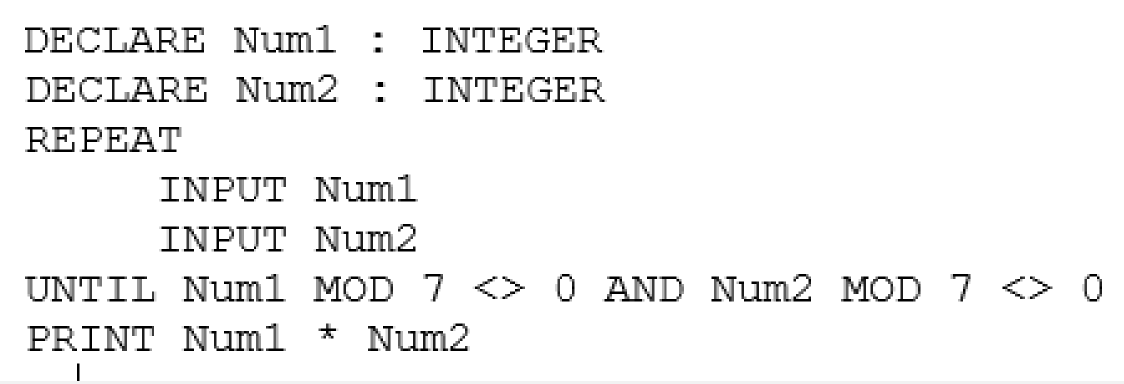
DECLARE ANY VARIABLES, CONSTANTS OR ARRAYS USED BY YOUR ALGORITHM.

6. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

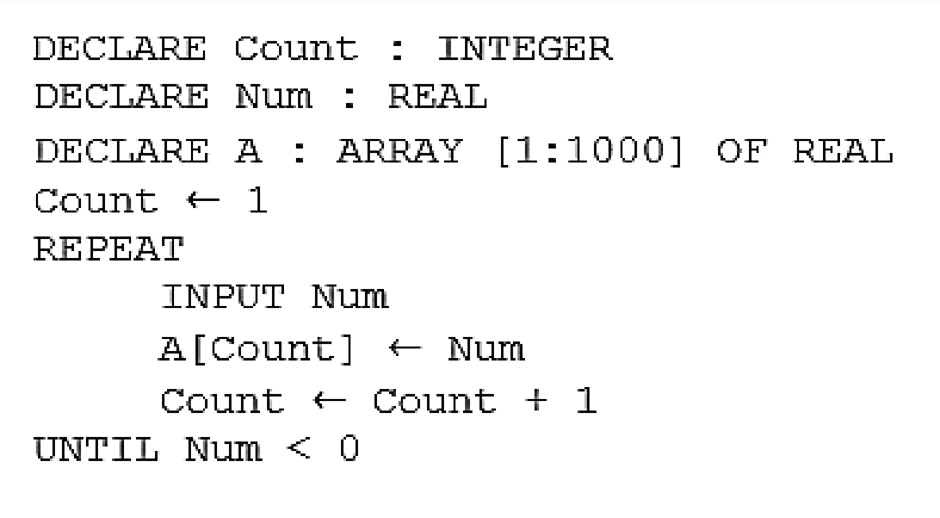
- ask the user to enter two whole numbers that are not multiples of 7
- stop asking for numbers when they enter two valid numbers
- calculate and output the product of the two valid numbers entered

7. Write an algorithm, using pseudocode containing a `REPEAT` loop, to:

- ask the user to enter numbers
- stop asking for numbers when the user enters a negative value
- store all the entered numbers into an array



```
DECLARE Num1 : INTEGER
DECLARE Num2 : INTEGER
REPEAT
      INPUT Num1
      INPUT Num2
UNTIL Num1 MOD 7 <> 0 AND Num2 MOD 7 <> 0
PRINT Num1 * Num2
```



```
DECLARE Count : INTEGER
DECLARE Num : REAL
DECLARE A : ARRAY [1:1000] OF REAL
Count ← 1
REPEAT
      INPUT Num
      A[Count] ← Num
      Count ← Count + 1
UNTIL Num < 0
```

THANK YOU

CREDIT

Credit to text, illustrations, images, videos, recordings, etc. owners in this presentation.

For educational purposes only.