# Worksheet 3 Iteration Answers

**Task 1**

1. Complete the trace table to determine the purpose of the following algorithm. Test it with input 14 and 5.

 OUTPUT ("Enter the first integer: ")

 x 🡨 USERINPUT

 OUTPUT ("Enter the second integer: ")

 y 🡨 USERINPUT

 z 🡨 0

 WHILE x > 0

 IF x mod 2 = 1 THEN

 z 🡨 z + y

 ENDIF

 x 🡨 x div 2

 y 🡨 y \* 2

 ENDWHILE

 OUTPUT ("Answer =", z)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **x** | **y** | **x mod 2** | **z** | **x > 0** | **output** |
| 14 | 5 | 0 | 0 | True |  |
| 7 | 10 | 1 | 10 | True |  |
| 3 | 20 | 1 | 30 |  |  |
| 1 | 40 | 1 | 70 | True |  |
| 0 | 80 |  |  | False | Answer = 70 |

The purpose of the algorithm is to multiply two integers.

It is known as the “Russian peasant’s algorithm”. Students can look it up on the Internet to find out why it works!

***See Python program W3 Q1 Russian peasants algorithm.py in folder.***

Get students to code the program and try it out with different integers.

2. A doctor records a patient’s temperature once an hour for six hours. Any time the temperature is > 37C, an incidence of fever is recorded.

 The average temperature is calculated at the end.

 (a) Calculate the expected result using test data 36, 36, 38.5, 37, 38, 36. Ans: 36.9

(b) Complete the trace table using the pseudocode below for this data.

temp 🡨 0

fever 🡨 0

total 🡨 0

hour 🡨 1 \*\*\*\*\*\* change to hour = 0

WHILE hour < 7 \*\*\*\*\*\* change to while hour < 6

 OUTPUT “Enter temperature: ”

 temp 🡨 USERINPUT

 IF temp > 37 THEN

 fever 🡨 fever + 1

 END IF

 total = total + temp

 hour = hour + 1

ENDWHILE

average 🡨 ROUND(total/hour,1) #round to 1 decimal place

OUTPUT “Average temperature:”, average

OUTPUT “Incidents of fever:”, fever

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **temp** | **fever** | **total** | **hour** | **average** | **Output** |
| 0 | 0 | 0 | 1 |  |  |
| 36 |  | 36 | 2 |  |  |
| 36 |  | 72 | 3 |  |  |
| 38.5 | 1 | 110.5 | 4 |  |  |
| 37 |  | 147.5 | 5 |  |  |
| 38 | 2 | 185.5 | 6 |  |  |
| 36 |  | 221.5 | 7 | 31.6 |  |
| (0) |  |  |  |  | “Average temperature: 31.6” |
|  |  |  |  |  | “Incidents of fever: 2” |

(c) Is the result correct? If not, make changes to the pseudocode so that it gives the correct result.

 No – need to make changes shown with \*\*\*\*\* above.

 ***See Python program W3 Q2 fever.py in folder.***

(d) Rewrite the pseudocode from the task above to include a range check to ensure that a temperature is between 30 and 44. Produce an error message for invalid data. The program should allow the user to re-enter the temperature if it is incorrect.

temp 🡨 0

fever 🡨 0

total 🡨 0

hour 🡨 0

WHILE hour < 6

 OUTPUT “Enter temperature”

 temp 🡨 USERINPUT

WHILE temp < 30 OR temp > 44

 OUTPUT “Invalid data: please re-enter”

 temp 🡨 USERINPUT

ENDWHILE

 IF temp > 37 THEN

 fever 🡨 fever + 1

 ENDIF

 total 🡨 total + temp

 hour 🡨 hour + 1

ENDWHILE

average 🡨 total/hour

OUTPUT “Average temperature: ”, average

OUTPUT “Incidents of fever ”, fever

**Task 2**

3. A parts supply company uses 4-digit part numbers. The last digit indicates the production run. If the production run is 6,7 or 8 it is considered to be an old model.

 Write a pseudocode algorithm that prompts the user to enter a part number.

 The length of the part number should be equal to 4 digits, otherwise an error message will be displayed and the user will be prompted to input the part number again.

 The algorithm should count the the total number of parts entered and the number of old model parts and output these totals.

 Data input will terminate when the user inputs 9999.

*Using a* ***WHILE*** *loop:*

oldModel 🡨 0

total 🡨 0

OUTPUT “Enter part num: ”

partnum 🡨 USERINPUT

WHILE partnum <> “9999”

 WHILE len(partnum) <> 4

 OUTPUT “Error: enter 4 digit number: ”

 partnum 🡨 USERINPUT

 ENDWHILE

 IF partnum[4] >= “6” AND partnum[4] <= “8” THEN

 oldModel 🡨oldModel + 1

 ENDIF

 total 🡨 total + 1

 OUTPUT “Enter part num: ”

 partnum 🡨USERINPUT

ENDWHILE

OUTPUT “Number of old models: ” , oldModel

OUTPUT “Total number of parts”, total

*or using a* ***REPEAT*** *loop:*

oldModel 🡨 0

total 🡨 0

REPEAT

 OUTPUT “Enter part num: ”

 partnum 🡨 USERINPUT

 IF partnum <> “9999” THEN

 IF len(partnum) <> 4 THEN

 REPEAT

 OUTPUT “Error: enter 4 digit number: ”

 partnum 🡨 USERINPUT

 UNTIL len(partnum) = 4

 ENDIF

 IF partnum[4] >= “6” AND partnum[4] <= “8” THEN

 oldModel 🡨 oldModel + 1

 ENDIF

 total 🡨 total + 1

 ENDIF

UNTIL partNum = “9999”

OUTPUT “Number of old models: ” , oldModel

OUTPUT “Total number of parts”, total

 ***See Python program W3 Q3 part numbers.py in folder.***

4, What is a common cause of an accidental infinite loop?

Forgetting to allow the data that will affect the termination value to change inside the loop, either through a calculation or through reading in user data.

**Task 3**

5. A teacher has a class of 30 pupils. Each pupil has taken 3 tests during the year. The teacher needs to know the average class score for test1, test2 and test3. She also needs to know the overall average test score for the year. Write an algorithm in pseudocode that will allow the teacher to input all results and print this information.

test1 🡨 0

test2 🡨 0

test3 🡨 0

total1🡨 0

total2 🡨 0

total3 🡨 0

totalYear 🡨 0

FOR index 🡨 1 to 30

 test1 🡨 USERINPUT

 test2 🡨 USERINPUT

 test3 🡨 USERINPUT

 total1 🡨 total1 + test1

 total2 🡨 total2 + test2

 total3 🡨 total3 + test3

NEXT index

totalYear 🡨 total1 + total2 + total3

average1 🡨 total1/30

average2 🡨 total2/30

average3 🡨 total3/30

averageYear 🡨 totalYear/30

OUTPUT “Averages:”, average1, average2, average3, averageYear

6. A Hallowe’en display needs a computer controlled light which will flicker. Flicker the light for a random number of seconds between 1/10 and 1/100 of a second. You can use a **pause** function that takes as a parameter the number of milliseconds to pause the program. For example **pause(1000)** will pause the program for 1 second. To turn the light on and off set the value of light to HIGH for ON and LOW for OFF. The control loop should run continuously.

x = 1

WHILE x = 1

 light 🡨 HIGH

 pause (random (10, 100))

 light 🡨 LOW

 pause (random (10, 100))

END WHILE

***See Python program W3 Q6 halloween.py in folder.***