Task 1: software design and development (part A)

AutoFeed has designed an automatic dog food dispenser. The product is designed to help dog owners prevent overfeeding. The food dispenser has five containers. Each container releases food, one at a time, at regular intervals throughout the day. The containers can hold a maximum of 200g of food.

AutoFeed wants an app to allow users to enter information about the size of their dog and the weight of food they will give them. The app uses this data to show if the weight of food is suitable for the size of dog.

AutoFeed research recommends the following weight of food (per day), depending on the size of dog.

Size of dog	Recommended weight of food per day
Small	From 110g to 140g
Medium	From 330g to 440g
Large	From 690g to 900g

After analysing the problem, the following inputs, processes and outputs have been identified.

Inputs

- the weight of food in each container
- the size of dog (small, medium or large)

Processes

- calculate the total weight of food in the five containers
- store a message that states if the total weight of food is within the recommended range
- calculate the average weight of food in the five containers
- round the average weight of food to one decimal place

Outputs

- the weight of food in each container
- the average weight of food in the five containers
- the rounded average weight of food
- the stored message stating if the total weight of food is within the recommended range

Assumptions

- the user can enter a weight using whole numbers or real numbers
- the user does not need to fill all five containers
- the user will not refill any of the containers until all containers are empty

Program design (pseudocode)

Main steps

1. Enter valid weight of food in each container and calculate total weight
2. Enter size of dog
3. Store a message that states if the total weight of food is within the recommended range
4. Calculate average weight
5. Display output messages

Refinements

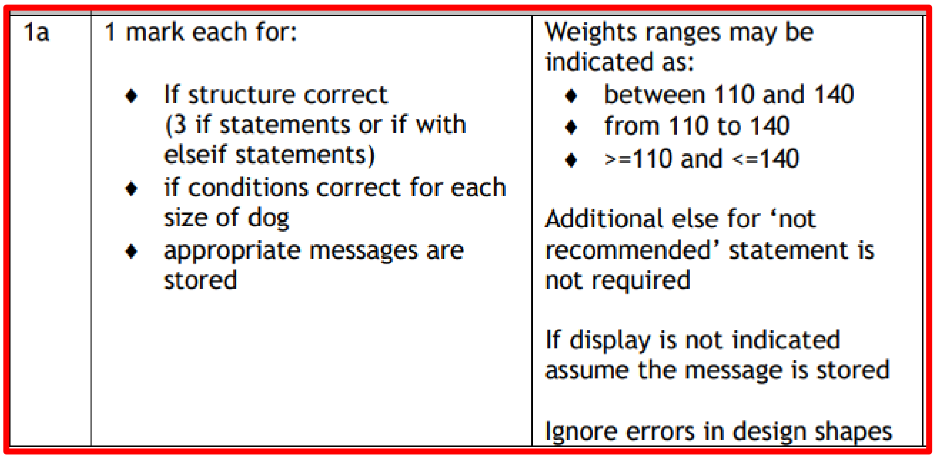
```
1.1     totalWeight = 0
1.2     start fixed loop 5 times
1.3         enter the foodWeight
1.4         while the foodWeight < 0 or foodWeight > 200
1.5             display “Invalid, a single container can only hold up to 200g”
1.6             re-enter the foodWeight
1.7         end while
1.8         totalWeight = totalWeight + foodWeight
1.9     end fixed loop

2.1     display “Please enter the size of your dog: small, medium or large”
2.2     enter size of dog

4.1     averageWeight = totalWeight / 5
4.2     round averageWeight to 1 decimal place

5.1     start fixed loop 5 times
5.2         display next foodWeight
5.3     end fixed loop
5.4     display total weight message
5.5     display average weight message
5.6     display recommendation message
```

1a Step 3 of the program design is incomplete:

3 Store a message that states if the total weight of food is within the recommended range

Using the information provided in the program analysis, expand the design to show how this process could be carried out. You can use a flowchart, structure diagram or pseudocode design.

(3 marks)



1a	1 mark each for: ♦ If structure correct (3 if statements or if with elseif statements) ♦ if conditions correct for each size of dog ♦ appropriate messages are stored	Weights ranges may be indicated as: ♦ between 110 and 140 ♦ from 110 to 140 ♦ >=110 and <=140 Additional else for 'not recommended' statement is not required If display is not indicated assume the message is stored Ignore errors in design shapes

```
3.1     if dog size = small and totalWeight is from 110 to 140 then
3.2         store message "This weight of food is suitable for your small dog."
3.3     else
3.4         if dog size = medium and totalWeight is from 330 to 440 then
3.5             store message "This weight of food is suitable for your medium dog."
3.6         else
3.7             if dog size = large and totalWeight is from 690 to 900 then
3.8                 store message "This weight of food is suitable for your large dog."
3.9             else
3.10                store message "This weight of food is not
                    recommended for the size of dog"
3.11            end if
3.12        end if
3.13    end if
```

□ Check your answers carefully, as you cannot return to part A after you hand it in.

□ When you are ready, hand part A to your teacher or lecturer and collect part B.

Candidate name__________________________ Candidate number__________________________

Task 1: software design and development (part B)

Program design (completed pseudocode)

Main steps

```
1 Enter valid weight of food in each container and calculate total weight 2
Enter size of dog
3   Store a message that states if the total weight of food is within the  recommended
    range
4   Calculate average weight
5   Display output messages
```

Refinements

```
1.1   totalWeight = 0
1.2   start fixed loop 5 times
1.3       enter the foodWeight
1.4       while the foodWeight < 0 or foodWeight > 200
1.5           display “Invalid, a single container can only hold up to 200g”
1.6           re-enter the foodWeight
1.7       end while
1.8       totalWeight = totalWeight + foodWeight
1.9   end fixed loop

2.1   display “Please enter the size of your dog: small, medium or large”
2.2   enter size of dog

3.1   if dog size = small and totalWeight is from 110 to 140 then
3.2       store message “This weight of food is suitable for your small dog.”
3.3   else
3.4       if dog size = medium and totalWeight is from 330 to 440 then
3.5           store message “This weight of food is suitable for your medium dog.”
3.6       else
3.7           if dog size = large and totalWeight is from 690 to 900 then
3.8               store message “This weight of food is suitable for your large dog.”
3.9           else
3.10                              store message “This weight of food is not
                      recommended for the size of dog”
3.11          end if
3.12      end if
3.13  end if

4.1   averageWeight = totalWeight / 5
4.2   round averageWeight to 1 decimal place

5.1   start fixed loop 5 times
5.2       display next foodWeight
5.3   end fixed loop
5.4   display total weight message
5.5   display average weight message
5.6   display recommendation message
```

1b Using the program analysis and the design, implement the program in a language of your choice.

Make sure the program matches the pseudocode provided on page 14.

(15 marks)

Print evidence of your program code.

Solution

https://trinket.io/python/85e2189c3d

1c (i) Test your program to make sure it produces the expected output.

Use the following test data to check that the message “This weight of food is suitable for your medium dog” is displayed:

Weight 1: 134.23
Weight 2: 74.99
Weight 3: 25.31
Weight 4: 112.33
Weight 5: 53.78
Size of dog: medium

Print evidence of the test run showing all inputs and the message displayed.

(1 mark)

(ii) Additional test data is required to check that the correct output messages are displayed.

Complete the test table below to show the expected results for Test 1 and appropriate inputs for Test 2.

(2 marks)

	Type of test	**Expected results**
Test 1	Weight 1: 30 Weight 2: 50 Weight 3: 45 Weight 4: 150 Weight 5: 70 Size of dog: large	This weight of food is not recommended for the size of dog
Test 2	Weight 1: 20 Weight 2: 30 Weight 3: 40 Weight 4: 10 Weight 5: 20 Size of dog:	"This weight of food is suitable for your small dog." Test 2 Inputs - Weights 1 to 5 should total between 110 and 140 size of dog should be small

Candidate name________________________ Candidate number___________________________

1d With reference to your code, evaluate your program by commenting on the following:

Efficiency of your program code **(2 marks)**

1 mark each for:

- identifying efficient code in own program
- justify why it is efficient

Efficiency examples could include:

- use of array
- nested ifs
- running total inside same loop as inputs
- use of a loop

Robustness of your completed program **(1 mark)**

1 mark:
how robust the program is, including if it copes with unexpected inputs with examples

Readability of your code **(1 mark)**

readability – comment on one aspect of readability in the candidate's own code

Evaluation must contain an element of evaluation rather than simple statements of terms.

For example "I have used white space to highlight structures in my program" not "I have used white space". The candidate's code must also show evidence of this for a mark to be awarded.

Candidate name_________________________ Candidate number________________________