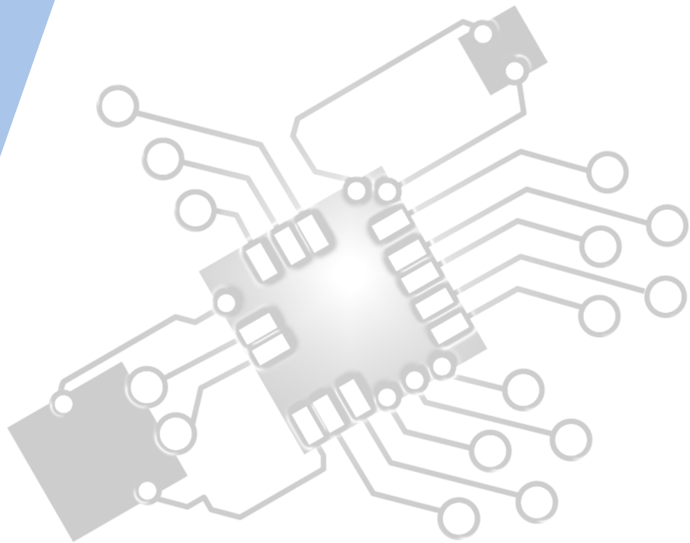
Computational thinking, problem-solving and programming:

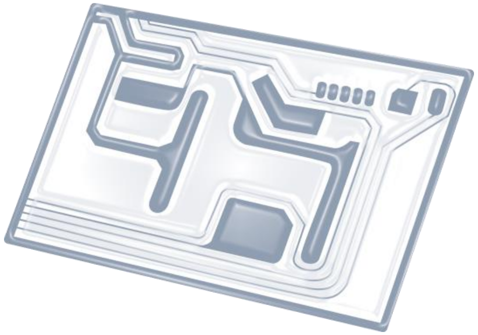
Connecting computational thinking and program design

IB Computer Science



Content developed by
Dartford Grammar School
Computer Science Department

HL Topics 1-7, D1-4



1: System design

2: Computer Organisation

3: Networks

4: Computational thinking

5: Abstract data structures

6: Resource management

7: Control

D: OOP

HL & SL 4.2 Overview

4.2.1 Describe the characteristics of standard algorithms on linear arrays

4.2.2 Outline the standard operations of collections

4.2.3 Discuss an algorithm to solve a specific problem

4.2.4 Analyse an algorithm presented as a flow chart

4.2.5 Analyse an algorithm presented as pseudocode

4.2.6 Construct pseudocode to represent an algorithm

4.2.7 Suggest suitable algorithms to solve a specific problem

4.2.8 Deduce the efficiency of an algorithm in the context of its use

4.2.9 Determine the number of times a step in an algorithm will be performed for given input data

1: System design

2: Computer Organisation

3: Networks

4: Computational thinking

5: Abstract data structures

6: Resource management

7: Control

D: OOP

Topic 4.2.7

Suggest suitable algorithms to solve a specific problem

Teacher’s notes:

- *Suitable algorithms may include **both standard algorithms** and **novel** algorithms.*
- *Suitable may include considerations of **efficiency**, **correctness**, **reliability** and **flexibility**.*
- *Students are expected to suggest algorithms that **will actually solve the problem successfully**.*

Content developed by Dartford Grammar School Computer Science Department

The faceplate of a car stereo has six buttons for selecting one of six preferred radio stations. As part of the internal representation of a microprocessor there is an array with six positions, carrying the information about the radio frequencies, as follows.

Radio

[0]	[1]	[2]	[3]	[4]	[5]
100.4	88.7	90.2	104.5	93.8	106.2

(a) State the information at `Radio[2]`. *[1]*

(b) Outline how a numerical frequency could be stored in a fixed-length string. *[2]*

(c) Construct an algorithm, in pseudocode, that calculates the range of frequencies (*ie* the difference between the highest and lowest frequencies) of any set of six selected radio stations. *[6]*

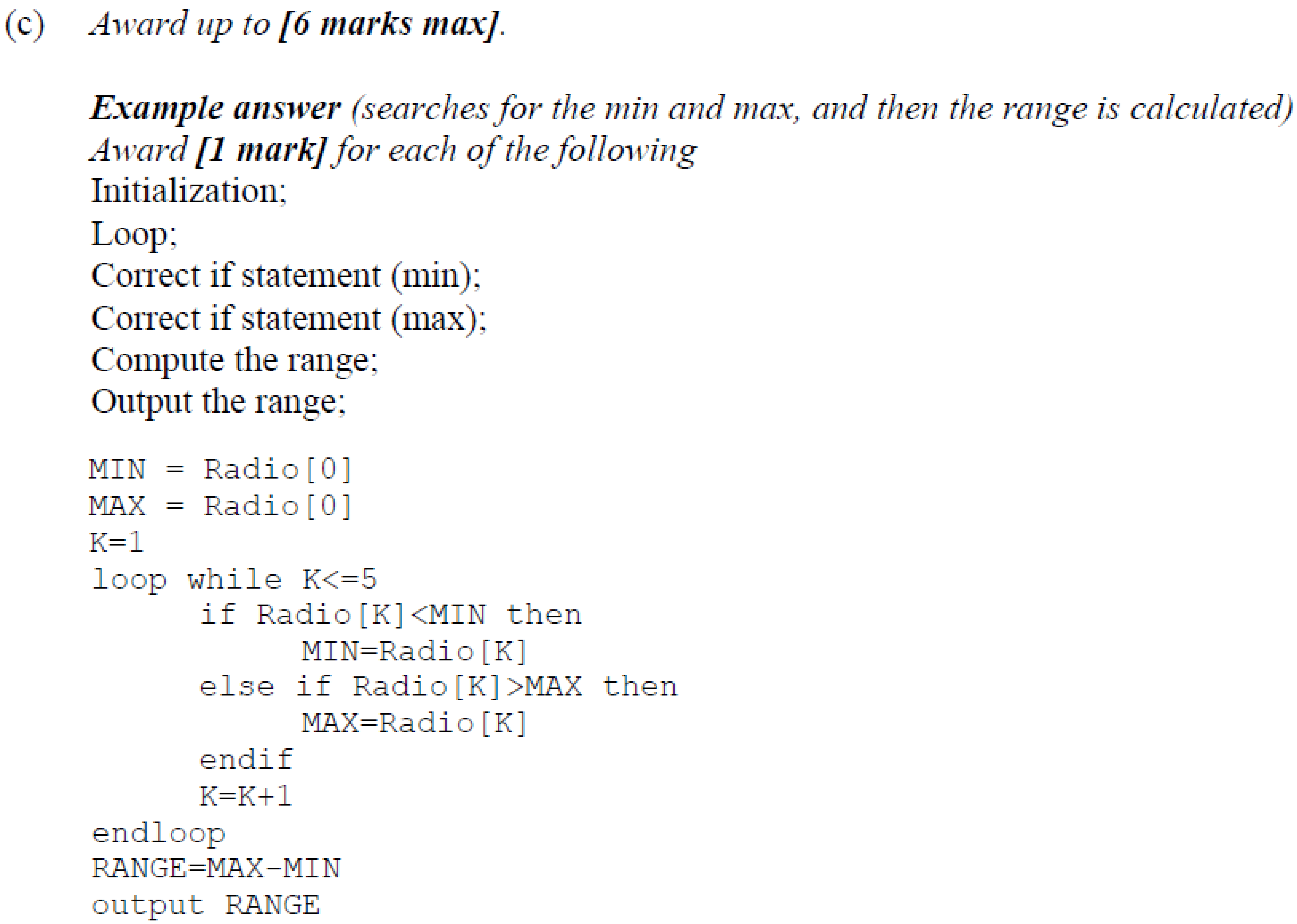
A display in the faceplate shows the name and frequency of the selected radio station. The name is automatically captured when storing a preference.

(d) Outline how a collection of objects could be used to store the name and frequency data in the radio. *[2]*

(e) Construct an algorithm, in pseudocode, to access and display the name and frequency of a station when a button is pressed. *[4]*

(a) 90.2; ***[1 mark]***

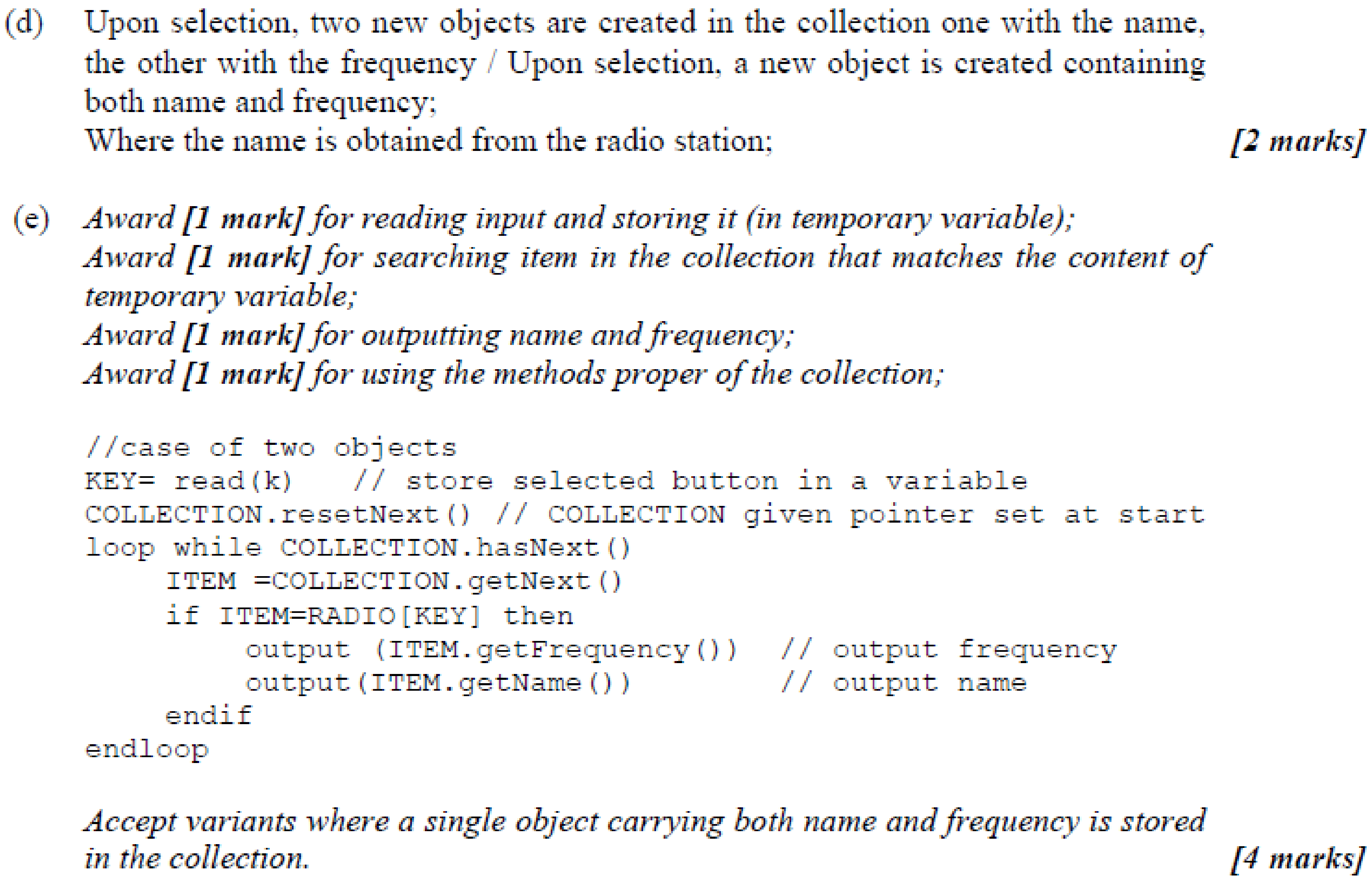
(b) Frequencies less than 100 take a 0 on the left (*eg* 88.7 becomes 088.7);
Convert each digit into a char to get a string;
Allow the "dot" to be omitted in the interpretation. There is always only one decimal in the example. ***[2 marks]***



(c) *Award up to **[6 marks max]**.*

***Example answer** (searches for the min and max, and then the range is calculated)*
*Award **[1 mark]** for each of the following*
Initialization;
Loop;
Correct if statement (min);
Correct if statement (max);
Compute the range;
Output the range;

```
MIN = Radio[0]
MAX = Radio[0]
K=1
loop while K<=5
      if Radio[K]<MIN then
            MIN=Radio[K]
      else if Radio[K]>MAX then
            MAX=Radio[K]
      endif
      K=K+1
endloop
RANGE=MAX-MIN
output RANGE
```



(d) Upon selection, two new objects are created in the collection one with the name, the other with the frequency / Upon selection, a new object is created containing both name and frequency;
Where the name is obtained from the radio station; ***[2 marks]***

(e) *Award* ***[1 mark]*** *for reading input and storing it (in temporary variable);*
Award ***[1 mark]*** *for searching item in the collection that matches the content of temporary variable;*
Award ***[1 mark]*** *for outputting name and frequency;*
Award ***[1 mark]*** *for using the methods proper of the collection;*

```
//case of two objects
KEY= read(k)    // store selected button in a variable
COLLECTION.resetNext() // COLLECTION given pointer set at start
loop while COLLECTION.hasNext()
     ITEM =COLLECTION.getNext()
     if ITEM=RADIO[KEY] then
          output (ITEM.getFrequency())   // output frequency
          output(ITEM.getName())         // output name
     endif
endloop
```

Accept variants where a single object carrying both name and frequency is stored in the collection. ***[4 marks]***

Tips for getting practice

- Try to come up with **your own problems** and see if you can solve them.
- Try to **share your created problems** with a friend and see if you can solve his/hers.
- For example:

  If PRICES and NAMES and INVENTORY are parallel arrays, write an algorithm that finds all the items where INVENTORY is below 10 items, and adds 20% to the PRICES of those items.