4.12.2 Writing Functional Programs

1) In a functional programming language, a function named *square* and three lists *a, b* and *c* are defined as follows.

$$square\ x\ =\ x\ *\ x$$
$$a\ =\ [2, 4, 6]$$
$$b\ =\ [3, 6, 9, 12, 15]$$
$$c\ =\ [8, 1, 5]$$

a) What is the list or value that is the result of applying the functions $head(tail(tail\ b))$?

[1 mark]

- **9**

b) Calculate the results of making the function calls listed in **Table 1** with the lists a, b and c above.

Table 1

Function Call	Result
$map\ square\ a$	$[4, 16, 36]$
$filter\ (<10)\ b$	$[3, 6, 9]$
$fold\ (+)\ 0\ c$	14

[3 marks]

2) In a functional programming language, four functions named fw, fx, fy and fz and a list named sales are defined as follows.

$$fw\ [a, b] = a * b$$
$$fx\ c = map\ fw\ c$$
$$fy\ d = fold\ (+)\ 0\ d$$
$$fz\ e = fy\ (fx\ e)$$
$$sales = [[8, 2], [1, 30], [4, 10]]$$

The sales list represents all the sales made in a shop in 1 day. It is composed of sublists.

The values in each sublist indicate the price of a product and the quantity of the product that was sold. For example, [8,3] indicates that 8 units of a product priced at £3 were sold.

a) **How many** of the four functions use a higher-order function?

[1 mark]

- **2**

b) Calculate the results of making the function calls listed in **Table 2**.

Table 2

Function call	Result
fw [7, 4]	**21**
fx sales	**[16, 30, 40]**
fz sales	**86**

[3 marks]

c) In the context of the shop, explain what the result of the function call fz sales represents.

[1 mark]

- **Total/one day's sales revenue/value/income**

3) The list *towers* is defined as:

$$towers = [\text{“}Blackpool\text{”}, \text{“}Paris\text{”}, \text{“}New\ Brighton\text{”}, \text{“}Toronto\text{”}]$$

a) State the head and the tail of this list.

[1 mark]

- **Head: “Blackpool” Tail: [“Paris”, “New Brighton”, “Toronto”]**

b) **Figure 1** shows some code written in a functional programming language.

Figure 1

$$total\ [\,] = 0$$
$$total\ (x{:}xs) = x + total\ (xs)$$

Describe how the *total* function works to add up all of the numbers in a list.

[3 marks]

- **The function is recursive**
- **It splits the list up into the head and the tail**
- **It calls itself with the tail of the list that it was called with**
- **Each call adds the value that is the head of the list to the total of the values in the tail of the list**
- **The recursion terminates when the list is empty by returning 0**

Functional programming languages support higher-order functions such as *map* and *fold*.

c) Explain what a higher-order function is.

[2 marks]

- **A function that takes a function as an argument…**
- **And/or returns a function as a result**

d) What is the result of this application of the *fold* function?

$$fold\ (*)\ 1\ [2, 3, 2]$$

[1 mark]

- **12**

4) In a functional programming language, six functions are defined in **Figure 2**.

Figure 2

```
temps = [50, 68, 95, 86]
fu a = (a - 32) * 5 / 9
fv b = map fu b
fw [ ] = 0
fw (x:xs) = 1 + fw (xs)
fx [ ] = 0
fx (x:xs) = x + fx (xs)
fy c = fx (c) / fw (c)
fz d = fy (fv (d))
```

A temperature can be converted from degrees Fahrenheit to degrees centigrade using the following method:

Centigrade = (Fahrenheit – 32) x 5/9

a) Which of the function in **Figure 2** includes a higher-order function in its definition.

[1 mark]

- **fv**

b) Which **two** of the functions in **Figure 2** use recursion in their definitions.

[1 mark]

- **fw and fx**

c) Calculate the results of making the function calls listed in **Table 2** using the functions in **Figure 2**.

Table 2

Function Call	Result
`fu 50`	**10.0 (accept integer 10)**
`fv temps`	**[10.0, 20.0, 35.0, 30.0] (accept integers)**
`fw temps`	**4**
`fz temps`	**23.75**

[4 marks]

d) Explain the purpose of the function fz.

[1 mark]

- **Calculates the average temperature in centigrade**

e) It is proposed that the definition of the function *fz* is changed to:

$$fz\ d\ =\ fu\ (fy\ (d))$$

Explain why this new definition of fz could be considered to be an improvement over the definition of fz in **Figure 2**.

[1 mark]

- **Only one conversion is done from Fahrenheit to centigrade // fewer conversions are performed // the function fv is no longer required**

5) In a functional programming language, six functions are defined in **Figure 3**.

Figure 3

$$FunctionZ\ [\ ]\ =\ 0$$
$$FunctionZ\ (x{:}\,xs)\ =\ x\ +\ 2\ *\ FunctionZ\ (xs)$$

a) Complete the table below by writing the value of the argument passed to each call of $FunctionZ$ and the value return by each call, when $FunctionZ\ [4, 2, 5, 3]$ is evaluated.

Call number	Argument	Value returned
1	[4, 2, 5, 3]	**52**
2	**[2, 5, 3]**	**24**
3	**[5, 3]**	**11**
4	**[3]**	**3**
5	**[]**	**0**

[3 marks]

b) All the values in lists passed to $FunctionZ$ as the argument are members of the set of integers. State what set is the co-domain of the function.

[1 mark]

- **The set of integers**

6) In a functional programming language, a recursively defined function named map and a function named $double$ are defined in **Figure 4**.

Figure 4

$$map\ f\ [\,]\ =\ [\,]$$
$$map\ f\ (x{:}\,xs)\ =\ f\ x\ :\ map\ f\ xs$$
$$double\ x\ =\ 2\ *\ x$$

a) In **Table 3**, write the value(s) that are the head and tail of the list [1, 2, 3, 4].

Table 3

Head	**1**
Tail	**[2, 3, 4]**

[1 mark]

b) Calculate the result of making the function call:

$$map\ double\ [\,1, 2, 3, 4\,]$$

[1 mark]

- **[2, 4, 6, 8]**

c) Explain how you arrived at your answer and the recursive steps that you followed.

[3 marks]

- **Explaining that map applies the function double to each list element**
- **Explaining that map applies double to the head of the list**
- **and then a recursive call is made on the tail of the list**