SL Unit 4 – Problem Solving

Quiz 2

Question 1			
Objectives:	4.1.18	Exam Reference:	May-14 11

Explain why an object is an example of abstraction. [2]

*Award up to **[2 marks max]**.*
An object hides the details;
Yet preserves the functionality;
OR
Objects combine abstractions of data and code;
While hiding away implementation of details;

Question 2			
Objectives:	4.2.5	Exam Reference:	Nov-15 6

A sub-program *all_even()* accepts a positive integer N and outputs *true* if all digits of N are even, otherwise it outputs *false*. For example, *all_even(246)* outputs *true* and *all_even(256)* outputs false.

The following algorithm is constructed for the sub-program *all_even(N)*.

```
EVEN = true
loop while (N > 0) and (EVEN = true)
     if (N mod 10)mod 2 = 1 then
          EVEN = false
     end if
end loop
output EVEN
```

(a) Explain why this algorithm does not obtain the correct result. [2]

*Award up to **[2 max]**.*
The value of N is never changed;
So the logical expression in the while loop always evaluates to true;
And loop repeats an infinite number of times;

(b) Outline what should be changed in the algorithm to obtain the correct result. [3]

Statement N = N div 10;
Should be written within the while loop;
After the if statement;

Question 3

Objectives:	4.2.1, 4.2.3, 4.2.6	Exam Reference:	Nov-15 9

1. A candy company manufactures 20 different kinds of candy, each identified by a product ID. An array, *Product_ID*, is used to store the product IDs, and another array, *Unit_Price*, is used to store the price per unit of each type of candy. The unit price of the product identified by *Product_ID[N]* is equal to *Unit_Price[N]* for any index *N*.

Product_ID		Unit_Price
Mints-1A	[0]	15.20
Choco-1B	[1]	18.10
Jelly-1Q	[2]	16.30
	…	
Choco-2A	[19]	11.90

(a) State the price of the candy identified by *Product_ID[2].* [1]

16.30;

(b) Explain the steps that would be needed in an algorithm to calculate the average unit price. [3]

Set a variable (sum) to zero;
Loop through the array `Unit_Price`;
Add each array element to variable sum;
Divide sum by 20;

(c) Construct the algorithm that will output the price of a candy after its product ID is entered by the user. The algorithm should output an appropriate message if the product ID entered does not appear in the array *Product_ID*. [6]

*Award marks as follows up to **[6]**.*
*Award **[1]** for the input.*
*Award **[1]** for introducing the Boolean variable.*
*Award **[2]** for the correct loop, **[1]** for a minor error.*
*Award **[1]** for the correct comparison in if statement.*
*Award **[1]** for correct assignment.*
*Award **[1]** for the output of the price after the loop.*
*Award **[1]** for 'does not appear' message.*

Example answer:

```
NUM = ENTERED ID
K = 0
FOUND = false
loop while K<20 and NOT FOUND
  if NUM = Product_ID[K] then
    FOUND = true
    PRICE = Unit_Price[K]
  end if
  K=K+1
end loop
if FOUND then
  output "The price is ", PRICE
else
  output NUM, " does not appear on the list of product numbers"
end if
```

The company maintains two warehouses each of which stocks a selection of the 20 types of candy indicated above.

The first warehouse stocks 15 items and their IDs are stored in an array, *One*. The second warehouse stocks 10 items and their IDs are stored in an array, *Two*.

All product IDs common to both warehouses will be placed in an array, *Three*.

(d) (i) State the maximum number of common product IDs which can be placed in *Three*. [1]

10

(ii) Construct the algorithm that will place all product IDs common to both warehouses in *Three*. [4]

*Award marks as follows up to **[4]**.*
*Award **[1]** for variable(Z) that keeps track of current position in array `Three`.*
*Award **[1]** for the correct outer loop.*
*Award **[1]** for the correct inner loop.*
*Award **[1]** for the condition.*
*Award **[1]** for the correct assignment.*

Example answer:

```
Z = 0
loop K from 0 to 14
   loop J from 0 to 9
      if One[K] = Two[J] then
         Three[Z] = One[K]
         Z=Z+1
      end if
   end loop
end loop
```

Question4			
Objectives:	4.2.1, 4.2.6, 4.2.7	Exam Reference:	Nov-17 13

A character array *S* holds the word "*PSEUDOCODE*".

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
P	S	E	U	D	O	C	O	D	E

(a) State the index of character "U" in the array *S*.

3

(b) Consider the following algorithm. The function *len()* returns the number of characters in an array (for example*, len(S)* is 10).

```
K = 0
CL = 0
loop while K < len(S)
  if S[K] = "E" then
    CL = CL + 1
  end if
  K = K + 1
end loop
output CL
```

For this algorithm, complete the following trace table. [4]

K	CL	K < len(S)	S[K] = "E"	output
0	0	TRUE	FALSE	…
…	…	…	…	…

*Award **[1]** for each correct column, other than column K.*

K	CL	K < len(S)	S[K] = "E"	output
0	0	TRUE	FALSE	
1	"	"	"	
2	1	"	TRUE	
3	"	"	FALSE	
4	"	"	"	
5	"	"	"	
6	"	"	"	
7	"	"	"	
8	"	"	"	
9	2	"	TRUE	
10	"	FALSE		2

***Note**: The symbol " appearing in cells above means that the value in that cell is the same as the one in the cell just above it; unfilled cells means that no value must be present.*

***Remark**: Accept drawings of tables where the values of the column CL are shifted downwards of one cell only, therefore showing the first value 2 when K = 10. Do not propagate by FT to the other columns, they are independent from the way CL has been filled.)*

A simple method of encoding a message is to use substitutions to produce a cryptogram.

Given a positive integer *N* and the array *UPCASELETTERS* containing letters in alphabetical order, a new array *SUBSTITUTE* is created by shifting the entire contents of *UPCASELETTERS* to the left, *N* times. As an element moves off the left of the array, it moves back into the right side of the array.
For example, given the array *UPCASELETTERS*:

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	[24]	[25]
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

When *N = 5* the array *SUBSTITUTE* will be:

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	[22]	[23]	[24]	[25]
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E

(c) Construct an algorithm which creates the array *SUBSTITUTE*. You may assume that a positive integer *N* and array *UPCASELETTERS* are given. [5]

Example answer 1
*Award marks as follows up to **[5 max]**.*
*Award **[1]** for correct boundaries in first for-loop.*
*Award **[1]** for correct assignment of array `SUBSTITUTE` in first loop.*
*Award **[1]** for correct boundaries in second for-loop.*
*Award **[1]** for correct assignment of array `SUBSTITUTE` in second loop.*
*Award **[1]** for incrementing `IND` in both loops.*

```
IND = 0
loop for K from N to len(UPCASELETTERS)-1
          // accept 'K from 0 to 25'
     SUBSTITUTE[IND] = UPCASELETTERS[K]
     IND = IND+1
end loop
loop for K from 0 to N-1
     SUBSTITUTE[IND] = UPCASELETTERS[K]
     IND = IND+1
end loop
```

This encoding method produces a cryptogram of a sentence by replacing each uppercase letter of the sentence with its substitute. Other characters in the sentence are not changed.

For example, using the arrays shown on page 6:

Input (sentence): ARS LONGA, VITA BREVIS.

Output (cryptogram): FWX QTSLF, ANYF GWJANX.

The following algorithm fragment inputs the characters, one by one, from the input sentence, and outputs its cryptogram using the method *encode().*

```
loop while NOT end-of-input-sentence
     CH = input()
     CRYPTEDCH= encode(CH, UPCASELETTERS, SUBSTITUTE)
     output CRYPTEDCH
end loop
```

The method *encode()* accepts a character *CH* and two arrays *UPCASELETTERS* and *SUBSTITUTE*, as defined above, and returns the corresponding character *CRYPTEDCH* of the character *CH*.

(d) Explain the steps to construct an algorithm for the method *encode().* [5]

*Award up to **[5 max]**.*

Search the array `UPCASELETTERS`;
Using a linear/sequential or binary search;
Search for the position/index of character `CH`;
If `CH` is found in `UPCASELETTER`;
`Return` the value stored in the array `SUBSTITUTE` at this position/index;
Otherwise `return CH`;