GCE AS

B500U20-1

O20-B500U20-1

TUESDAY, 13 OCTOBER 2020 – AFTERNOON

COMPUTER SCIENCE – AS component 2
Practical Programming to Solve Problems

2 hours 15 minutes

B500U201
01

INSTRUCTIONS TO CANDIDATES

Answer **ALL** of questions 1, 2 and 3.

Answer only **ONE** section of question 4. This is the section which requires you to use the Integrated Development Environment (IDE) of your chosen programming language.

You will need to record all of your answers to questions 1, 2 and 3 in a **single** word processed document.

INFORMATION FOR CANDIDATES

The number of marks is given in brackets at the end of each question or part-question.

You are reminded of the need for good English and orderly, clear presentation in your answers.

The total number of marks available is 60.

You will need a computer with an installed functional copy of the Integrated Development Environment (IDE) appropriate to your chosen programming language and word processing software.

A calculator is allowed in this examination.

Remember to save your work regularly.

Scenario

Metropolis
Apartment Rentals

Metropolis Apartment Rentals (Met) is a small, independent apartment rental company. The company has many apartments to rent to customers. Once a potential customer moves into an apartment they are referred to as a tenant.

The manager of **Met** has decided to commission a new computerised system to store **Met** staff details, apartment details and tenant details such as tenant ID, first name, surname and contact details.

The main relationships of the system are:

- One member of staff manages many apartments, an apartment is always managed by a single named staff member.
- An apartment is rented to one or more tenants at any time. A tenant may only rent one apartment at any given time.

You have been commissioned to develop a prototype computer system.

1. *(a)* **Met** would like to create an entity relationship diagram for the main relationships in the system. Referring to the scenario create an entity relationship diagram for **Met's** system. There is no need to add any attributes. [8]

(b) **Met** has decided to store tenant details in a database table. Referring to the original scenario copy and complete the following data structure table. [4]

Fieldname	Key field (Yes/No)	Data Type	Field Length	Possible Validation

2. **Met** is considering an object-oriented approach to programming its computer systems. It wishes to use a class diagram to describe the relationships between the classes in part of its payment processing system.

Met would like a superclass called `Transaction`.

The `Transaction` class should have two protected attributes: `accountNumber` and `transactionID`, which are of type string and integer respectively. The `Transaction` class should have two public methods to get each of the two attributes, which both return a single item of the same data type as stored.

Met would like a subclass called `AddCredit` which inherits from class `Transaction`.

The `AddCredit` class should have a private attribute called `amount` of type real. The `AddCredit` class should have a public method for setting `amount` which accepts a parameter of type real.

Create a class diagram for this situation. [8]

3. *(a)* **Met** wishes to model the potential increase in sales arising should it offer each tenant a discount for introducing **Met** to other potential customers. **Met** has the following algorithm which attempts to model the increase in sales.

```
1   Begin Subroutine sales()
2   declare a as integer
3   declare b as integer
4   declare week as integer
5   declare targetSales as integer
6   declare iteration as integer
7
8   set week = 0
9   set a = 1
10  set b = 1
11  set iteration = 1
12
13  repeat
14
15   output iteration
16
17   set week  = a
18   output week
19
20   targetSales = a + b
21   output targetSales
22
23   set a = b
24   set b = targetSales
25   set iteration = iteration + 1
26
27  until iteration = 5 {next loop}
28
29  End Subroutine
```

Copy and complete the table to show the outputs of the algorithm. [8]

iteration:	Week:	Target Sales:
1		
2		
3		
4		

3. *(b)* **Met** are considering using an array to store tenant data. They will need to search for items within the array. The array will not be sorted as data will be added weekly.

Using a recognised convention, design an algorithm to help **Met** search the contents of an array for a specific item.

The algorithm should allow a user to input the item to be searched and return the message "Found" as well as the location in the array if the data is found. The algorithm should also return a suitable message if the data is not found within the array.

Your algorithm should be written using self-documenting identifiers. [8]

4. Select the programming language of your choice from section ***(a)***, ***(b)*** or ***(c)*** and answer **all** questions in your chosen section.

(a) **Visual Basic**

Met wants a computer system to be developed using **Visual Basic** that will be used to:

- Store apartment and tenant details
- Recall and count specific apartment or tenant details
- Store and recall specific details input by the user.

(i) Open the file MetApartment
- Read through the code and familiarise yourself with its contents
- The file contains incomplete code, which is intended to allow **Met** to store and count apartments with specific details.

Complete this code. [4]

Remember to save the changes made to the file MetApartment

(ii) Using the internal facility of the IDE, add annotation to the code from question 4*(a)*(i) that would clearly explain the design of the program to another software developer. [4]

Save your annotations in the same file as 4*(a)*(i) above.

(iii) Create a new form or program that will allow **Met** to:

- Input tenant details
- Validate tenant details
- Store tenant details on disk in a text file called tenantDetails.txt
- Confirm storage of the details
- Retrieve specified tenant details from disk. [12]

Save your work as MetTenants

(iv) Using the internal facility of the IDE, add annotation to your code from question **4*(a)***(iii) that would clearly explain the design of your program to another software developer. [4]

Save your annotations in the same file as 4*(a)*(iii) above.

(b) **Java**

Met wants a computer system to be developed using **Java** that will be used to:

- Store apartment and tenant details
- Recall and count specific apartment or tenant details
- Store and recall specific details input by the user.

(i) Open the file MetApartment
- Read through the code and familiarise yourself with its contents
- The file contains incomplete code, which is intended to allow **Met** to store and count apartments with specific details.

Complete this code. [4]

Remember to save the changes made to the file MetApartment

(ii) Using the internal facility of the IDE, add annotation to the code from question 4*(b)*(i) that would clearly explain the design of the program to another software developer. [4]
Save your annotations in the same file as 4*(b)*(i) above.

(iii) Create a new form or program that will allow **Met** to:

- Input tenant details
- Validate tenant details
- Store tenant details on disk in a text file called tenantDetails.txt
- Confirm storage of the details
- Retrieve specified tenant details from disk.

[12]

Save your work as MetTenants

(iv) Using the internal facility of the IDE, add annotation to your code from question 4*(b)*(iii) that would clearly explain the design of your program to another software developer. [4]

Save your annotations in the same file as 4*(b)*(iii) above.

(c) **Python**

Met wants a computer system to be developed using **Python** that will be used to:

- Store apartment and tenant details
- Recall and count specific apartment or tenant details
- Store and recall specific details input by the user.

(i) Open the file MetApartment
- Read through the code and familiarise yourself with its contents
- The file contains incomplete code, which is intended to allow **Met** to store and count apartments with specific details.

Complete this code. [4]

Remember to save the changes made to the file MetApartment

(ii) Using the internal facility of the IDE, add annotation to the code from question 4*(c)*(i) that would clearly explain the design of the program to another software developer. [4]

Save your annotations in the same file as 4*(c)*(i) above.

(iii) Create a new form or program that will allow **Met** to:

- Input tenant details
- Validate tenant details
- Store tenant details on disk in a text file called tenantDetails.txt
- Confirm storage of the details
- Retrieve specified tenant details from disk. [12]

Save your work as MetTenants

(iv) Using the internal facility of the IDE, add annotation to your code from question 4*(c)*(iii) that would clearly explain the design of your program to another software developer. [4]

Save your annotations in the same file as 4*(c)*(iii) above.

END OF PAPER

BLANK PAGE

BLANK PAGE