Section C

You are advised to spend no more than **15 minutes** on this section.

Type your answers to **Section C** into your Electronic Answer Document.

You **must save** this document at regular intervals.

These questions refer to the **Preliminary Material** and the **Skeleton Program**, but **do not** require any additional programming.

Refer **either** to the **Preliminary Material** issued with this question paper **or** your electronic copy.

0 6 This question is about the method `GetIndexOfCompany` in the `Simulation` class.

0 6 . 1 What does a return value of `-1` indicate?

[1 mark]

0 6 . 2 Some of the program code in this method is unnecessary. Identify which code is not necessary and explain why.

[2 marks]

0 7 This question is about the `Outlet` class.

0 7 . 1 State the name of a method in the `Outlet` class that uses string concatenation.

[1 mark]

0 7 . 2 State the name of a local variable in a method in the `Outlet` class.

[1 mark]

0 7 . 3 The `Outlet` class contains some protected attributes. Explain the difference between protected and private attributes.

[1 mark]

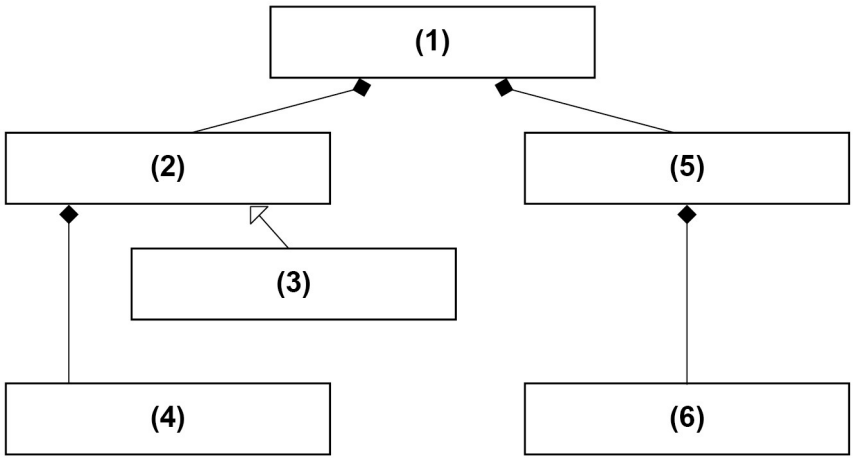
0 7 . 4 In the `Outlet` class the constant value `100` has been used when calculating the initial value of `DailyCosts` in the constructor method.

Explain what problem will occur if this value in the constructor is changed and no other changes are made.

[1 mark]

0 8 **Figure 4** shows a partially completed class diagram that describes the relationships between some of the classes used in the Skeleton Program.

Figure 4



0 8 . 1 State the identifier of the class that has been labelled **(3)** in **Figure 4**.

[1 mark]

0 8 . 2 State the identifier of the class that has been labelled **(4)** in **Figure 4**.

[1 mark]

0 8 . 3 State the identifier of the class that has been labelled **(5)** in **Figure 4**.

[1 mark]

0 8 . 4 Aggregation, composition and inheritance are three different types of relationship that can exist between classes.

Which of these three types of relationship is **not** shown in **Figure 4**?

[1 mark]

0 9 This question is about the `ProcessDayEnd` method in the `Simulation` class.

Explain how the program decides which company an individual household has decided to use when eating out.

[4 marks]

Turn over for Section D

Section D

You are advised to spend no more than **70 minutes** on this section.

Enter your answers to **Section D** in your Electronic Answer Document.

You **must save** this document at regular intervals.

These questions require you to load the **Skeleton Program** and to make programming changes to it.

1 0 This question refers to the method `AddCompany` in the `Simulation` class.

The program is to be changed so that it checks that the company name entered is valid. The company must have a name and it cannot be the same as the name of an existing company. The program should keep doing these checks until a valid name for the company has been entered.

What you need to do

Task 1
Modify the method `AddCompany` so that an appropriate error message is displayed if the user presses the Enter key without entering a company name when they are asked to enter the name of the company.

Task 2
Modify the method `AddCompany` so that an appropriate error message is displayed if the user enters the name of a company that is already being used for a company in the simulation when they are asked to enter the name of the company.

You may use the method `GetIndexOfCompany` to find out if a company name has already been used, though you do not have to use this method in your answer.

Task 3
Test that the changes you have made work:

- run the Skeleton Program
- choose to use a normal size settlement
- choose to use the default companies
- choose the option to add a new company from the menu
- press the Enter key without entering a name for the new company
- type in the name `AQA Burgers` and press the Enter key
- type in the name `In Jest` and press the Enter key.

Note: you will get three marks for your program code if either Task 1 or Task 2 has been successfully completed and you will get five marks if both Task 1 and Task 2 have been successfully completed.

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 0 . 1 Your PROGRAM SOURCE CODE for the amended method `AddCompany`.

[5 marks]

1 0 . 2 SCREEN CAPTURE(S) showing the results of the requested test.

[1 mark]

Turn over for the next question

1 1 This question extends the functionality of the simulation by adding a new type of household, an affluent household, that eats out considerably more often than other households.

What you need to do

Task 1
Create a new class called `AffluentHousehold` that inherits from the `Household` class. The constructor for this new class should make a call to the constructor of the `Household` class and then set the value of `ChanceEatOutPerDay` to 1

Task 2
Modify the `AddHousehold` method in the `Settlement` class so that it creates an `AffluentHousehold` object instead of a `Household` object if the randomly-generated value for `X` is less than 100

Task 3
Test that the changes you have made work:

- run the Skeleton Program
- choose to use a normal size settlement
- choose to use the default companies
- then choose the menu option to display the details of the households and check that there is a mixture of both types of household in the settlement.

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 1 . 1 Your PROGRAM SOURCE CODE for the amended method `AddHousehold` and the new class `AffluentHousehold`.

[7 marks]

1 1 . 2 SCREEN CAPTURE(S) showing the requested test.

[1 mark]

Turn over for the next question

1 2 This question extends the Skeleton Program by adding the facility for a company to obtain a loan to increase their balance.

When a company chooses to get a loan, 10 000 is added to their balance. The company should only be allowed to get a loan if they do not already have a loan that has not been paid back. The user will be asked to specify the daily interest rate for the loan.

The company can pay back all or part of the loan at any time. The program should keep track of the amount of the loan that the company still needs to pay back.

Each time the simulation is advanced by a day, the company will have their balance reduced by an amount equivalent to the interest rate applied to the amount of the loan minus any repayments that have been made. **Figure 5** shows an example of the calculation.

Figure 5

A company has repaid 1500 of the loan meaning that it still owes 8500. The interest rate for the loan was 0.0005% and the company's current balance is 4000. The simulation is advanced by a day and the company's balance is reduced by 0.0425 to 3999.9575 (0.0425 is 0.0005% of 8500).

To get full marks on this question you will need to make use of encapsulation to hide attributes in the `Company` class.

What you need to do

Task 1
Add attributes to the `Company` class named `LoanBalance` and `InterestRate`. `LoanBalance` should be used to track the amount of the loan that the company needs to pay back. `InterestRate` should be used to store the interest rate at which the loan was taken.

Task 2
Modify the `ModifyCompany` method in the `Simulation` class so that there are options on the menu to get a loan and to pay back a loan.

Task 3
Modify the Skeleton Program so that if a company has a loan balance of 0 then it can take out a loan. When it does:

- 10 000 should be added to `Balance` and `LoanBalance`
- `InterestRate` should be set to the value specified by the user.

Task 4
Modify the Skeleton Program so that when a company chooses to pay back a loan the user is asked how much to pay back and the values of `LoanBalance` and `Balance` are reduced by this amount. You **do not** need to check that the value entered is sensible or that the company has a loan to pay back.

Task 5
Modify the `ProcessDayEnd` method in the `Company` class so that the company's daily interest for the loan is subtracted from the company's balance.

Task 6
Test that the changes you have made work:

- run the Skeleton Program
- choose to use a normal size settlement
- choose to use the default companies
- give the company AQA Burgers a loan with a percentage interest rate of 0.015
- advance the simulation
- get the company AQA Burgers to pay back 500 of the loan
- advance the simulation
- then choose the menu option to display the details of the companies.

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 2 . 1 Your PROGRAM SOURCE CODE for the amended parts of your Skeleton Program. You should include the entire code for each method you have changed/created.

[10 marks]

1 2 . 2 SCREEN CAPTURE(S) showing the requested test. You only need to show the result of choosing the menu option to display the details of the companies. Evidence for the earlier parts of the test is not needed.

[1 mark]

Turn over for the next question

1 3 Currently, the Skeleton Program creates a delivery route for a company based on the order in which a company's outlets have been created – the route starts with the oldest outlet, then goes to the second oldest outlet and so on until all the outlets have been included in the route.

The route produced specifies the order in which the outlets are visited when making deliveries – the longer the route, the higher the delivery cost for the company.

This question extends the Skeleton Program by using a greedy algorithm to try to reduce the total distance of the delivery route. A greedy algorithm is one that takes the choice which seems to be the best each time; it is not guaranteed to find the optimal solution.

Figure 6 shows how the greedy algorithm should work.

Figure 6

The greedy algorithm should:

1. start by adding the oldest outlet (the one in position 0 in `Outlets`) to the route
2. then find the company's outlet that is nearest (has the smallest distance) to the outlet that was last added to the route, and which has not yet been added to the route, and add this to the end of the route
3. repeat step 2 until all the company's outlets have been added to the route.

Figure 7 shows how the delivery cost is arrived at for the company Paltry Poultry which has four outlets. The outlet numbers (0–3) refer to the index of an outlet within the `Outlets` data structure for that company. This example may help you check your code works correctly.

Figure 7

- The oldest outlet (outlet 0) is at coordinates (800,390) within the settlement; this will be the first outlet on the route.
- From this outlet the closest outlet is outlet 2 which is at (820,370) so outlet 2 will be added as the second outlet on the route.
- From outlet 2 the closest outlet not yet in the route is outlet 3 which is at (800,600) so outlet 3 will be added as the third outlet on the route.
- From outlet 3 the closest outlet not yet in the route is outlet 1 which is at (400,390) so outlet 1 will be added as the fourth outlet on the route.
- The correct order that the outlets should be visited is 0, 2, 3, 1
- The delivery cost would be 6.96708

What you need to do

Task 1
Create a new method `GetOrderedListOfOutlets` in the `Company` class. The new method should return a list that represents the route created using the algorithm in **Figure 6**. Each outlet should be represented in the list by the index of its position in `Outlets`.

Task 2
Modify the `CalculateDeliveryCost` method in the `Company` class so that it uses the list returned by the new method `GetOrderedListOfOutlets` instead of the list returned by the method `GetListOfOutlets`.

Task 3
Test that the changes you have made work:

- run the **Skeleton Program**
- choose to use a normal size settlement
- choose to use the default companies
- choose the menu option to display the details of the companies.

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 3 . 1 Your PROGRAM SOURCE CODE for the amended method `CalculateDeliveryCost` and for the new method `GetOrderedListOfOutlets`.

[11 marks]

1 3 . 2 SCREEN CAPTURE(S) showing the test described in **Task 3**. The screen capture should show the delivery cost for **AQA Burgers**.

[1 mark]

END OF QUESTIONS