M11/5/COMSC/HP2/ENG/TZ0/XX

22117012

International Baccalaureate®
Baccalauréat International
Bachillerato Internacional

COMPUTER SCIENCE
HIGHER LEVEL
PAPER 2

Friday 20 May 2011 (morning)

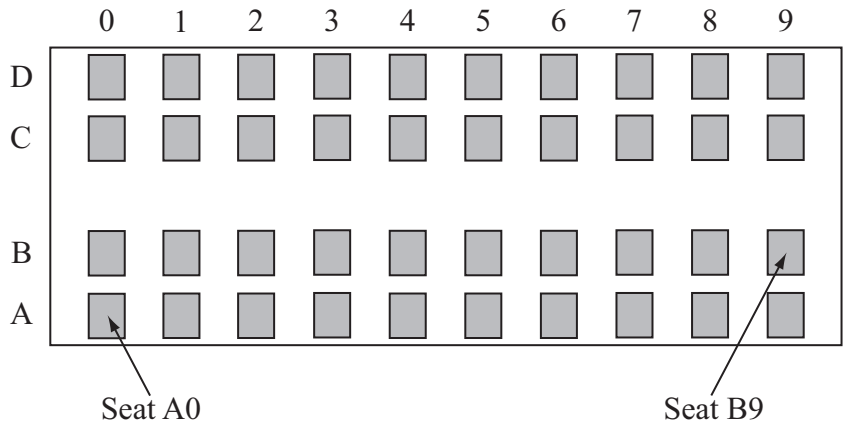
2 hours 15 minutes

INSTRUCTIONS TO CANDIDATES

- Do not open this examination paper until instructed to do so.
- Answer all the questions.

*Answer **all** the questions.*

1. A small airplane numbers its seats as shown in the diagram below.



A program has been written that allocates seats to passengers. In the program, the details for each seat are stored as a `Seat` object. The `Seat` class is partly defined below.

```
public class Seat
{
  private String seatName;       // e.g. "A0"
  private String passengerName; // e.g. "Jones" if reserved or
                                // null if not reserved

  public Seat    // constructor is incomplete

  // accessor methods
  public String getPassengerName() { return passengerName; }
  public String getSeatName()      { return seatName; }

  // transformer methods
  public void setPassengerName(String p) { passengerName = p; }
  public void setSeatName(String s)      { seatName = s; }
}
```

The two class variables are declared as **`private`**.

(a) Explain the feature of object-oriented programming (OOP) that can be implemented when class variables are declared as **`private`**. *[3 marks]*

(b) Complete the constructor for the `Seat` class (started above), that would allow a seat name and passenger name to be assigned to a `Seat` object. *[3 marks]*

(This question continues on the following page)

(Question 1 continued)

The class that manages the allocation of seats to the various passengers is called `SeatManager`. The `SeatManager` class, which uses objects of the `Seat` class, is partially shown below.

```
public class SeatManager
{
   ...seatArray... // seatArray is a two-dimensional array variable
                   // used to store details of all of the seats

   public int seatsRemaining()    // returns the number of empty seats
   {}

   public String freeWindowSeat() // defined in part(e) below
   {}
}
```

(c) Describe fully the two-dimensional array `seatArray` which will be used to store the seat names and passenger names of all of the seats. *[3 marks]*

As shown in the diagram, the airplane numbers its rows from 0 to 9. Each row contains 4 seats labelled A, B, C and D. Each seat can be identified by a letter and a number (*e.g.* "A0").

(d) By making use of nested loops and the array variable `seatArray`, construct the method `seatsRemaining()` that will return the number of seats that have **not** yet been allocated. *[5 marks]*

Window seats are those that are labelled "A" or "D".

(e) Construct a method that would return the `seatName` of a window seat that is available (has not already been allocated), or a suitable message if all windows seats have already been allocated. *[4 marks]*

A larger airplane has two levels (upper and lower) containing seats for the passengers.

(f) Suggest **one** way in which the data structure described in part (c) could be changed or modified to allow for this. *[2 marks]*

2. An unordered file contains records storing the names and year of birth of different people.

For example:

Molina, 1932; Lakhi, 1910; Arthurs, 1962; …

To print out the names in order of year of birth, the records are first read one by one into a linked list, so that the list is **always** correctly sorted in ascending order of year.

(a) By using a diagram, construct the linked list after the 3 records shown above have been read in. *[2 marks]*

(b) Explain, with the aid of a diagram, how the node *Guy, 1915* would now be added to the list. *[2 marks]*

The class `Node`, shown below, allows for new nodes to be created for the linked list.

```
public class Node
{
  public String name;
  public int yearOfBirth;
  public Node next;      // points to the next node in the list
                         // or to null if there is not one

  public Node(String x, int y, Node z)      // constructor
  {
    this.name = x;
    this.yearOfBirth = y;
    this.next = z;
  }
}
```

(c) Explain how a new `Node` object with the following data would be created.

`name`: Xiao
`yearOfBirth`: 1920
`next`: null *[2 marks]*

(This question continues on the following page)

(Question 2 continued)

The class `ListManager` contains methods that manage the list. The class is partially shown below.

```
public class ListManager
{
  private Node first; // points to the first node in the linked list

  // the method addNode() adds a node to the correct position in
  // the list so that the list remains sorted
  public void addNode(String name, int year)
  {
    // lines of code missing
  }

  // the method removeLeast() removes the first node in the list
  public void removeLeast()
  {
    // lines of code missing
  }
}
```

(d) Construct the method `removeLeast()` that has been started above. *[2 marks]*

(e) Construct the method `addNode()` that has been started above. *[8 marks]*

(f) Analyse the BigO efficiency of

(i) the `removeLeast()` method; *[2 marks]*

(ii) the `addNode()` method. *[2 marks]*

3. A very large file containing employee data, which is principally used for reference, is stored on disk. The program that manages this file allows a particular record to be searched for. The key field is the ID number of each person.

Various methods of file organization are being considered.

(a) (i) Outline the steps required to display the required record if the file organization was *sequential*. *[2 marks]*

(ii) Explain why it would be impractical to store this data as a sequential file. *[2 marks]*

An alternative is *direct access file organization*.

(b) (i) Explain how a *hashing algorithm* can be used to locate the required record on disk. *[4 marks]*

(ii) State **three** requirements of a well designed hashing algorithm. *[3 marks]*

It is decided to allow the user to search for a record by more than one field (*e.g.* the person's surname as well as the ID number).

(c) (i) Explain why directly using a hashing algorithm now becomes impractical. *[2 marks]*

(ii) Explain how indexing the file can allow searching by more than one field. *[3 marks]*

Each index will be read into a data structure in the memory.

(d) (i) Suggest reasons why a *dynamic* data structure might be used instead of a *static* data structure. *[2 marks]*

(ii) Explain why a *binary search tree* would be preferred to a *linked list* as the dynamic data structure. *[2 marks]*

4. *This question requires the use of the case study*.

(a) With reference to the diagram on page 3 of the case study, explain the advantage of networking the Air Traffic Control (ATC) system with the Flight Information Display System (FIDS). *[2 marks]*

(b) (i) Identify **one** communication medium which would be used to connect different parts of the airport network. *[1 mark]*

(ii) Suggest reasons for the communication medium identified in part (i). *[2 marks]*

(c) Describe a suitable security measure that could confirm the identity of employees as they pass through the various sections of the airport. *[2 marks]*

(d) Describe **two** features related to computer systems found inside a modern airport that improve the passenger experience. *[4 marks]*

(e) A modern airport relies on highly computerized systems. By making reference to **one** area of airport operation, discuss how this can be both an advantage and a disadvantage for the traveller. *[5 marks]*

(f) Suggest how tagging passengers inside an airport could

(i) improve airport security; *[2 marks]*

(ii) help the airlines maintain their schedules. *[2 marks]*

(g) (i) With reference to the baggage system on page 9 of the case study, suggest **two** ways in which the use of computer simulation in the development of the baggage system would have been beneficial to the design team. *[4 marks]*

(ii) Explain why maintenance is an important part of the system design cycle. *[3 marks]*

(h) Explain why encryption should be an important feature of Wi-Fi hotspots. *[3 marks]*

The passenger list for each flight would initially be stored in order of time of booking.

(i) Explain how a binary tree could be used to display all passenger details in alphabetical order of passenger surnames, without storing all of the passenger information in the main memory. *[4 marks]*

(This question continues on the following page)

(Question 4 continued)

The Departures screen displays the details of the next 11 flights that are scheduled to leave. After each departure the details of that flight are removed from the screen. Flights do not always leave at their scheduled time.

(j) (i) Describe a suitable data structure to hold these details in memory. *[3 marks]*

(ii) Explain the steps involved in changing the display after each departure. *[3 marks]*