1. (a) (i) *Award up to **[2 marks max]**.*
Queue is a FIFO data structure;
A list in which items may be added only at one end;
And removed only at the other end; ***[2 marks]***

(ii) *Award up to **[2 marks max]**.*
Stack is a LIFO data structure;
A list in which one of the ends is designated as the top of the stack;
And access (store and retrieve) is restricted to this end of the list; ***[2 marks]***

(b) *Example answers:*
Transfer of data from/to I/O devices;
Simulation;
Job queue, order of processing; ***[1 mark]***

(c) 5; ***[1 mark]***

(d) (i) The queue is empty! ; ***[1 mark]***

(ii) *Award marks as follows up to **[4 marks max]**.*
*Award **[1 mark]** if the queue is not empty;*
*Award **[1 mark]** for temporarily storing the value of the item;*
*Award **[1 mark]** if this is at the beginning of the queue;*
*Award **[1 mark]** for changing the pointer that points to the end of the queue if the item to be deleted is the last one;*
*Award **[1 mark]** for changing the value of the pointer that points to the beginning of the queue;*
*Award **[1 mark]** if it points to the next item in the queue;*
*Award **[1 mark]** for returning the value that was at the beginning of the queue;*

```
public int dequeue()
{
  if (isEmpty())
  {
    output ("Queue empty");
    return -1;
  }
  else
  {
    int temp=first.item;
    if (first.next== null) // only 1 item in queue
    { last = null;}
    first = first.next // first node removed
    return temp:
  }
}
```

[4 marks]

continued ...

Question 1 continued

(iii) *Award marks as follows up to **[3 marks max]**.*
*Award **[1 mark]** for deleted item 2.*
*Award **[1 mark]** for three items in the queue.*
*Award **[1 mark]** for all three correct values (4, 1, 7).*

Deleted item: 2
Items in the queue: 4, 1, 7 ***[3 marks]***

(e) *Award up to **[6 marks max]**.*
Initialize an empty stack;
While queue is not empty;
Remove the element from the beginning of the queue/dequeue;
Push the removed element onto the stack;
While stack is not empty;
Pop an element off the stack;
Display it/enqueue it;

Possible answer:
Take the items off the queue;
And put them one by one;
In a new;
Stack;
Now take them off the stack;
And put them back in the queue; ***[6 marks]***

Total: [20 marks]

2. (a) (i) 19; ***[1 mark]***

(ii) 13; ***[1 mark]***

(b) *Award **[1 mark]** for each correct output line, up to **[4 marks max]**.*
Total number of sales for Salesman 1 is 23;
Total number of sales for Salesman 2 is 19;
Total number of sales for Salesman 3 is 29;
Total number of sales for Salesman 4 is 31; ***[4 marks]***

(c) *Award marks as follows up to **[6 marks max]**.*
*Award **[1 mark]** for correct method heading.*
*Award **[2 marks]** for correct outer loop (**[1 mark]** for minor error).*
*Award **[1 mark]** for initializing total.*
*Award **[2 marks]** for correct inner loop (**[1 mark]** for minor error).*
*Award **[2 marks]** for correctly increasing total by `Sales[n][m]`, (**[1 mark]** for an attempt).*
*Award **[1 mark]** for output.*

Example answer:

```
public void myst2(int[][] Sales)
{
   for (int m = 0; m < 5; m = m + 1)
   {
      int total = 0;
      for (int n = 0; n < 4; n = n + 1)
      {
         total = total + Sales[n][m];
      }
      output("Total number of sales for Model " + (m + 1) + " is " + total);
   }
}
```
[6 marks]

continued ...

Question 2 continued

(d) *Award marks as follows up to **[8 marks max]**.*
*Award **[1 mark]** for correct method heading.*
*Award **[1 mark]** for initializing variables `highestAmount` and `bestSalesman`.*
*Award **[2 marks]** for correct outer loop (**[1 mark]** for minor error).*
*Award **[1 mark]** for initializing `totalAmount`.*
*Award **[2 marks]** for correct inner loop (**[1 mark]** for minor error).*
*Award **[2 marks]** for correct calculation of `totalAmount` (**[1 mark]** for minor error).*
*Award **[3 marks]** for if statement, (**[1 mark]** for condition, **[1 mark]** for each assignment statement (×**2**)).*
*Award **[1 mark]** for each output (×**2**).*

Example answer:

```
public void determineBest(int[][] Sales, double[] ModelPrice)
{
  double highestAmount = 0.0;
  int bestSalesman = 0;

  for (int z = 0; z < 4; z++)
  {
    double totalAmount = 0.0;

    for (int k = 0; k < 5; k++)
    {
      totalAmount = totalAmount + Sales[z][k] * ModelPrice[k];
    }

    if (totalAmount > highestAmount)
    {
      highestAmount = totalAmount;
      bestSalesman = z;
    }
  }
  output("The best salesman is Salesman " + (bestSalesman + 1));
  output("The highest total number of sales is " + highestAmount);
}
```

[8 marks]

Total: [20 marks]