*Answer **all** the questions.*

1. The following code represents a partial implementation of a stack.

```
public class Node
{
  private String name;
  private Node next;
  public Node()
  {
    name = null;
    next = null;
  }
  public void setName(String n){ name = n; }
  public void setNext(Node n){ next = n; }
  public String getName(){ return name; }
  public Node getNext(){ return next; }
}
public class Stack
{
  private Node top;
  public Stack()
  {
    top = null;
  }
  public void push(String name)
  {
    Node temp = new Node();
    temp.setName(name);
    temp.setNext(top);
    top = temp;
  }
}
```

(a) Outline the steps involved in executing the following statement.

```
Node temp = new Node();
```

[3 marks]

(b) Draw a diagram showing the structure and content of the stack nodes after the following statements have been executed.

```
Stack s = new Stack();
s.push("Lisa");
s.push("Creighton");
s.push("Annabel");
```

[3 marks]

(This question continues on the following page)

(Question 1 continued)

(c) Construct the method, `pop`, which pops a name from the stack. Return an error message and an empty String if the stack is empty. *[7 marks]*

An array, `names`, with six elements, contains the following data.

index	[0]	[1]	[2]	[3]	[4]	[5]
contents	"Elissa"	"Margaret"	"Shannon"	"Kevin"	"Joe"	"Sophie"

You are going to construct an algorithm which reverses the names in the array so that the new contents are as follows.

index	[0]	[1]	[2]	[3]	[4]	[5]
contents	"Sophie"	"Joe"	"Kevin"	"Shannon"	"Margaret"	"Elissa"

(d) Construct an algorithm that could be used to reverse the contents of this array using the Stack class. You may assume it contains a correct implementation of the method `pop`. *[3 marks]*

(e) Describe **two** other methods that could usefully be added to the Stack class. *[4 marks]*

*Answer **all** the questions.*

1. (a) In the context of *data structures*, explain what is meant by a

(i) *queue*; *[2 marks]*

(ii) *stack*. *[2 marks]*

(b) State **one** computer application for which a queue is a suitable data structure. *[1 mark]*

Consider the following class.

```
class Node
{
  public int item;
  public Node next;

  public Node(int d)
  {
    item = d;
    next = null;
  }

  public void displayNode()
  {
    output(item + " ");
  }
}
```

(c) Statement `Node x = new Node(5);` creates an object of class type `Node`. State the output produced by the call `x.displayNode();`. *[1 mark]*

(This question continues on the following page)

(Question 1 continued)

Examine the following linked list implementation of a queue.

```
class MyQueue
{ private Node first;
  private Node last;
  public MyQueue() { first = null;  last = null; }
  public boolean isEmpty() { return first == null; }

  public void enqueue(int x)
  { Node newNode = new Node(x);
    if (isEmpty())
    { first = newNode; }
    else
    { last.next = newNode; }
    last = newNode;
  }

  public int dequeue()
  {
    // Code missing that will remove a node from the queue
  }

  public void displayQueue()
  { if (first == null)
    { output("The queue is empty!"); }
    else
    { Node temp = first;
      while (temp != null)
      { temp.displayNode();
        temp = temp.next;
      }
    }
  }
}
```

(d) The statement `MyQueue x = new MyQueue();` creates an empty queue.

(i) State the output that will be produced after execution of the following statement.

```
x.displayQueue();
```

[1 mark]

(ii) Construct the code for the method `dequeue()`. The method should remove one item from `x` and return the value of the removed item. *[4 marks]*

(iii) State the output that will be produced after execution of the following statements.

```
x.enqueue(2);
x.enqueue(4);
int y = x.dequeue();
output("Deleted item: " + y);
x.enqueue(1);
x.enqueue(7);
output("Items in the queue: ");
x.displayQueue();
```

[3 marks]

(e) Explain how the elements in a non-empty queue could be reversed using a stack. *[6 marks]*