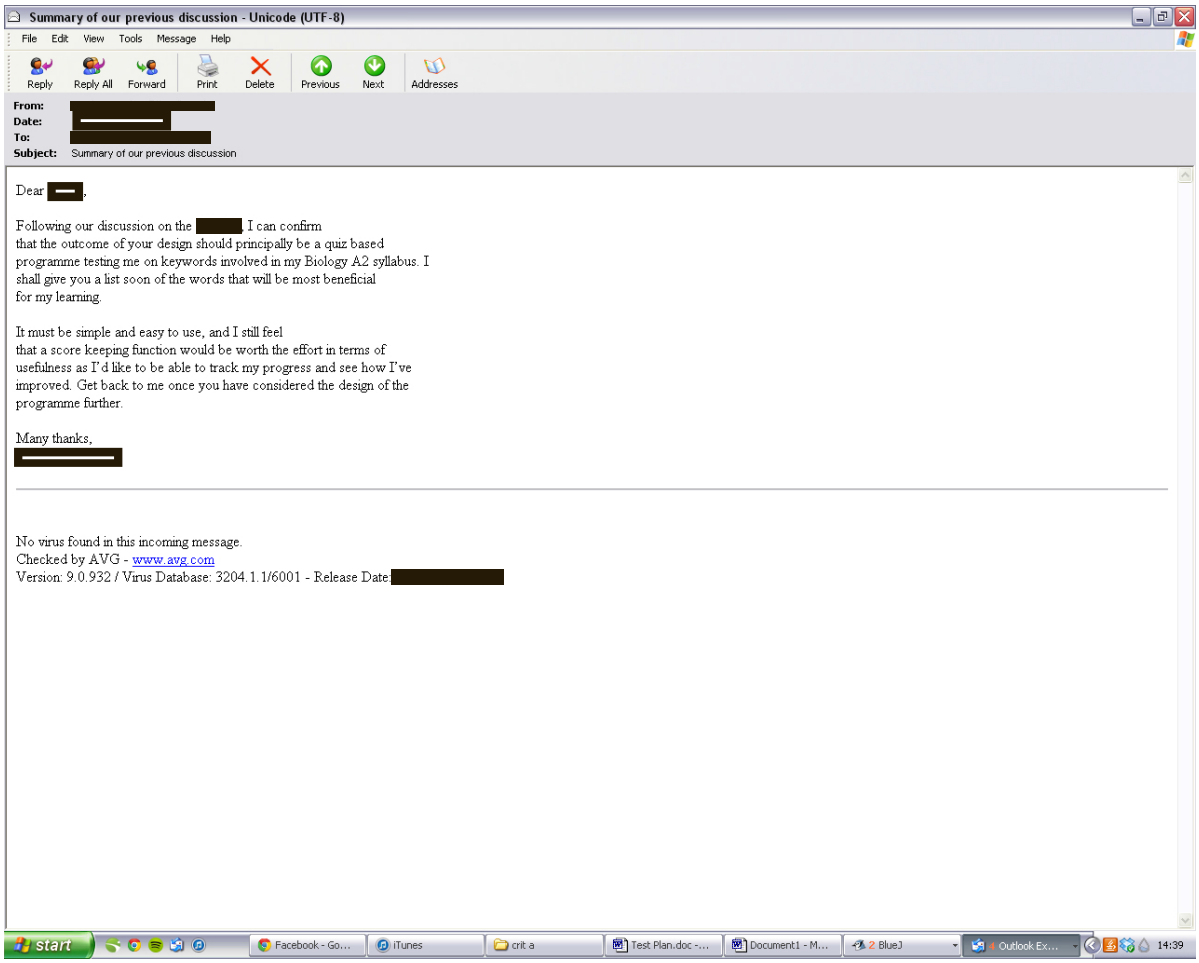
Appendix

Dear xx,

Following our discussion on the xx, I can confirm
that the outcome of your design should principally be a quiz based
programme testing me on keywords involved in my Biology A2 syllabus. I
shall give you a list soon of the words that will be most beneficial
for my learning.

It must be simple and easy to use, and I still feel
that a score keeping function would be worth the effort in terms of
usefulness as I'd like to be able to track my progress and see how I've
improved. Get back to me once you have considered the design of the
programme further.

Many thanks,
xx



Notes from meeting with client: xx

- Client confirmed that the program could be text based: so long as simplicity was maintained.
- Client agreed list of definitions – confirmed that 45 was a sufficient number.
- Client brought up random nature of the quizzes – that this was important.
- Agreed that scores be given at the end of each quiz for simplicity.
- Seemed pleased with aspects of design so far – flowchart.
- Client also agreed that functionality should be improved to prevent repeated questions.

Notes from meeting with client: xx

- Early version of 'finished' product shown to client – quiz and menu working.
- Client happy with design so far, confirmed that 'text based' was okay for the quiz, at least.
- Pointed out some mistakes in the wording of the questions etc. Need to be checked and tested.
- Otherwise pleased with progress, but would have liked implementation of persistent storage of scores so that she could gauge her progress.
- I agree to try and implement this within the program.

Notes from final meeting with client: 15/12/13

- Client showed slight misgiving about final design, but came round once she understood simplicity of it
- Pleased with quiz function, and found it worked effectively, being quite testing!

- Enjoyed random function and lack of repeated questions, as well as ability to set the length of the quiz
- Pleased that it could be used simply, only requiring installation of java and use of a console. No extra expense!
- Overall, concluded that the program would be very useful for her revision, and she would start using it immediately.
- We both agreed that the program had achieved the success criteria.

Source Code

Menu Class

```
import java.util.*;
import java.util.Scanner;
import java.io.IOException;

public class Menu
{
   private Quiz quiz;
   private Database database;
   private Scanner menuInput;
   private String firstInput;

   /**
    * Constructor for objects of class Menu
    */
public static void main (String[] args) throws IOException {
      Menu menu = new Menu();
   }
   public Menu() throws IOException
   {
      // clears the screen
      System.out.print("\f");
      // used to check input
      menuInput = new Scanner(System.in);
      firstInput = "placeholder";
      // should the user input close, the program closes
      while (!firstInput.equals("close")) {
         System.out.println("What would you like to do? Type 'quiz' to use the quiz, type 'print' to
print the database of definitions, 'scores' to see your previous scores and 'close' to close the
program.");
         // takes the user input as a string
         firstInput = menuInput.next();
         if (firstInput.equalsIgnoreCase("quiz")){
            // user sets length of the quiz
            System.out.println("How long would you like the quiz? (Maximum 45 and input whole
numbers)");
            String s = menuInput.next();
            Integer x;
            // ensures that user input is an integer, if not requests a different input.
            try
            {
               x = new Integer(s);
            }
            catch(Exception e)
            {
               System.out.println("Please input a whole number under 45");
               continue;
            }
            quiz = new Quiz(x);
```

```
        }
        // prints definitions for the user
        if (firstInput.equalsIgnoreCase("print")){
            database = new Database();
            database.printArray();
        }
        if (firstInput.equalsIgnoreCase("scores")){
            FileData file = new FileData();
            file.readFile();
        }
      }
      // closes program
      if (firstInput.equalsIgnoreCase("close")){
         System.out.print("\f");
         return;
      }}

   public String getFirstInput()
   {
      return firstInput;
   }

   public void setFirstInput(String newFirstInput)
   {
      firstInput = newFirstInput;
   }
}
```

Quiz Class

```
import java.util.*;
import java.util.Scanner;
import java.util.Random;


public class Quiz
{
   private int questionNumber;
   private int quizScore;
   private int quizLength;
   private Question question;
   private Scanner userInput;
   private String input;
   private boolean questionMatch;
   private int randomQuestionNumber;
   private String currentQuestion;
   private String currentAnswer;
   // creates the list used to prevent repeated questions
   private List<String> askedQuestions = new ArrayList<String>();
   // creates the database object from which the questions are read
   private Database database = new Database();
```

```
    public Quiz(int quizLength)
    {
        // ensure that a quiz larger than the database of questions cannot be created
        if (quizLength <= database.getArrayLength())
        {
            // clears the text window
            System.out.print("\f");
            questionNumber = 1;
            // allows the creation of the first question in createQuestion
            questionMatch = true;
            quizScore = 0;
            this.quizLength = quizLength;
            // used for evaluating user input
            Scanner userInput = new Scanner(System.in);
            // creates the arrays within the database class
            database.setArrays();
            // runs through the quiz to the length which the user asked for.
            while (questionNumber < quizLength + 1 ){
                createQuestion();
                askedQuestions.add(currentQuestion);
                this.question.printQuestion();
                System.out.print("What is this defining? Enter your answer. Enter close to close the
program. ");
                String input = userInput.nextLine();
                if (input.equalsIgnoreCase("close")){
                    return;
                }
                checkAnswer(input, question);
                questionNumber++;
                questionMatch = true;
            }
            printScore();
            // Writes score to file
            String score = quizScore + " out of " + quizLength;
            FileData file = new FileData();
            try{
                file.writeToFile(score);
            }
            catch (Exception e)
            {
                System.out.println(e.getMessage());
            }
        }
        // an error message returning the user to the 'menu' if they try to create a quiz that is too long
        else {
            System.out.println("Error, the quiz length must be under " + database.getArrayLength() + " .");
        }
    }

    public void printScore()
```

```
    {
        System.out.println("Your Score is: " + quizScore + " out of " + quizLength);
    }

    public int getScore()
    {
        return quizScore;
    }

    public int getLength()
    {
        return quizLength;
    }

    public int getQuestionNumber()
    {
        return questionNumber;
    }

    public void checkAskedQuestions(int size, String question)
    {
        int i = 0;
        questionMatch = false;
        // iterates through the list of questions already asked, comparing them to the question about to
be asked to ensure no repeats.
        while (i < size && questionMatch == false && size > 0 )  {
            String tempQuest = askedQuestions.get(i);
            if(tempQuest.equals(question)) {
                // ends the 'while' loop, returning to createQuestion to ensure the creation of a new
question.
                questionMatch = true;
            }
            else {
                i++;
            }
        }
    }

    public void createQuestion() {
        while(questionMatch == true) {
            // creates a new class of type 'Random'
            Random randomCreator = new Random();
            // uses the nextInt method to generate a random integer between 0 (inclusive) and 45
(exclusive)
            int randomQuestionNumber = randomCreator.nextInt(45);
            int n = askedQuestions.size();
            // ensures that the question has not already been used
            checkAskedQuestions(n, database.getQuestion(randomQuestionNumber));
            if (questionMatch == false) {
                // accesses the question and answer and creates a new question object with them
                currentQuestion = database.getQuestion(randomQuestionNumber);
```

```
            currentAnswer = database.getAnswer(randomQuestionNumber);
            question = new Question(currentQuestion, currentAnswer, questionNumber);
        }
      }
    }

    public void checkAnswer(String userAnswer, Question currentQuestion)
    {
      // retrieves the answer from the question currently being asked
      String a = currentQuestion.getAnswer();
      // checks to see if the strings are equal (ignoring capital letters), marks accordingly.
      if (userAnswer.equalsIgnoreCase(a)){
        System.out.println("Correct!") ;
        quizScore = quizScore+1;
      }
      else {
        System.out.println ("Wrong! The correct answer was " + a);
      }

    }
}
```

Question Class

```
import java.util.*;
import java.util.Scanner;

public class Question
{
    private String question;
    private String answer;
    private int questionNumber;

    public Question(String currentQuestion, String currentAnswer, int questionNumber)
    {
      // assigns the variables for Question class
      this.question = currentQuestion;
      this.answer = currentAnswer;
      this.questionNumber = questionNumber;
    }

    public void printQuestion()
    {
      System.out.println(questionNumber + ". " + question);
    }

    public String getAnswer()
    {
      return answer;
    }
```

```
    public String getQuestion()
    {
        return question;
    }

    public void setAnswer(String newAnswer)
    {
        answer = newAnswer;
    }

    public void setQuestion(String newQuestion)
    {
        question = newQuestion;
    }

}
```

Database Class

```
import java.util.*;

public class Database
{
    private String[] questionsArray;
    private String[] answersArray;
    private int arrayLength;

    /**
     * Constructor for objects of class Database
     */
    public Database()
    {
        setArrays();
        arrayLength();
    }
    // Creates The questions and answers arrays.
    public  void setArrays() {

        answersArray = new String[45];

        answersArray[0] = "abiotic";
        answersArray[1] = "acetycholine";
        answersArray[2] = "action potential";
        answersArray[3] = "actin";
        answersArray[4] = "activation energy";
        answersArray[5] = "active transport";
        answersArray[6] = "adenosine triphosphate";
        answersArray[7] = "adrenaline";
        answersArray[8] = "aerobic";
        answersArray[9] = "osmosis";
```

```
        answersArray[10] = "exocytosis";
        answersArray[11] = "endocytosis";
        answersArray[12] = "diffusion across membranes";
        answersArray[13] = "facilitated diffusion across membranes";
        answersArray[14] = "hydrophilic";
        answersArray[15] = "Hydrophobic";
        answersArray[16] = "Peripheral Proteins";
        answersArray[17] = "Integral Proteins";
        answersArray[18] = "Glycoproteins";
        answersArray[19] = "Receptor Proteins";
        answersArray[20] = "Primary";
        answersArray[21] = "Secondary";
        answersArray[22] = "Tertiary";
        answersArray[23] = "Peptide bond";
        answersArray[24] = "Ester bond";
        answersArray[25] = "Glycosidic bond";
        answersArray[26] = "Width of a magnified object";
        answersArray[27] = "Valid";
        answersArray[28] = "Reliability";
        answersArray[29] = "Precision";
        answersArray[30] = "Accuracy";
        answersArray[31] = "Dependent Variable";
        answersArray[32] = "Independent Variable";
        answersArray[33] = "Gene";
        answersArray[34] = "Allele";
        answersArray[35] = "Homologous";
        answersArray[36] = "Homozygous";
        answersArray[37] = "Heterozygous";
        answersArray[38] = "Carrier";
        answersArray[39] = "Genotype";
        answersArray[40] = "Phenotype";
        answersArray[41] = "Recessive";
        answersArray[42] = "Dominant";
        answersArray[43] = "Covalent bond";
        answersArray[44] = "Hydrolysis";




        questionsArray = new String[45];
        questionsArray[0] = "An ecological factor that makes up part of the non-biological environment
of an organism.";
        questionsArray[1] = "A neurotransmitter (transmitter substance) found in cholinergic
synapses.";
        questionsArray[2] = "A brief reversal of the resting potential across the cell surface membrane
of a neurone.Gives a value of +40 mV.";
        questionsArray[3] = "A protein found in muscle cells. It is the main component of the thin
filaments.";
        questionsArray[4] = "Energy required to bring about a chemical reaction. This is lowered by the
presence of enzymes.";
```

```
    questionsArray[5] = "Movement of a substance across a membrane from a region where it is in low concentration to a region where it is in a high concentration";
    questionsArray[6] = "An activated nucleotide found in all living cells that acts an an energy carrier.";
    questionsArray[7] = "A hormone produced by the adrenal glands in times of stress that prepares the body for an emergency.";
    questionsArray[8] = "Connected with the presence of oxygen. This type of respiration requires oxygen to release energy from glucose and other foods";
    questionsArray[9] = "The net movement of water molecules frokm a solution with a lower concentration of solute (high conc. of water) to a solution with a higher concentration of solute (low conc. of water) across a partially permeable membrane.";
    questionsArray[10] = "The release of substances (usually proteins or polysaccharides) from the cell as vesicles (small membrane-bound sacs) which fuse with the cell surface membrane, releasing their contents.";;
    questionsArray[11] = "The absorption of substances (usually proteins or polysaccharides) into a cell by the creation of a vesicle. Part of the cell membrane engulfs the substance to be transported.";
    questionsArray[12] = "The net movement of molecules or ions from a region where they are at a higher concentration to a region of their lower concentration, until equilibrium is reached. ____ across membranes.";
    questionsArray[13] = "Channel proteins are used to aid large hydrophilic molecules and ions that cannot diffuse ordinarily. Each channel protein has a specific shape to allow only one type of molecule or ion.";
    questionsArray[14] = "Water-attracting";
    questionsArray[15] = "Water-repelling";
    questionsArray[16] = "Proteins that are loosely attached on the outside of a membrane";
    questionsArray[17] = "Proteins that are fully embedded inside a membrane phospholipid bilayer.";
    questionsArray[18] = "A molecule consisting of a protein and a carbohydrate chain. It is used for cell recognition, to make cells form together to form tissues.";
    questionsArray[19] = "Allow the cell to react to other cells and hormones";
    questionsArray[20] = "A chain of amino acids joined by peptide bonds - Polypeptides. The ____ structure of a protein.";
    questionsArray[21] = "When the polypeptide chain starts to curl up due to interactions between the different side groups of the amino acids that it is connecting. Contains alpha-helix (coil) and beta-pleated sheets (zig-zags). Both are held together with hydrogen bonds.The ____ structure of a protein";
    questionsArray[22] = "The three-dimensional shape caused after all sections of the polypeptide chain have curled or twisted.The ____ structure of a protein.";
    questionsArray[23] = "The bond between two amino acids";
    questionsArray[24] = "The bond between the glycerol and the fatty acid tail in a phospholipid. (Phosphoester bond between phosphate group and glycerol)";
    questionsArray[25] = "The bond between the monosaccharides to form carbohydrates";
    questionsArray[26] = "actual measurement= measured size / magnification";
    questionsArray[27] = "When something supports the idea, it is said to be... ";
    questionsArray[28] = "When results from a repeated experiment are similar they show the experiment's... ";
    questionsArray[29] = "When a repeated experiment gives a similar result every time";
    questionsArray[30] = "How close to the true value a reading/measurement is.";
    questionsArray[31] = "The variable that relies on the result of the experiment";
    questionsArray[32] = "The fixed variable that the experimenter may have control over";
    questionsArray[33] = "A section of DNA that codes for a polypeptide";
```

```
        questionsArray[34] = "A different form of the same gene";
        questionsArray[35] = "The chromosomes in each of pair of the 23 pairs in each cell.";
        questionsArray[36] = "Identical alleles (HH)";
        questionsArray[37] = "Two different alleles (Hh)";
        questionsArray[38] = "Someone with both a mutated gene and a normal gene, but the mutated
gene is recessive.";
        questionsArray[39] = "The two alleles that a person has (HH, Hh or hh)";
        questionsArray[40] = "The appearance of a characteristic in an organism (cystic fibrosis/ not
cystic fibrosis)";
        questionsArray[41] = "The allele that only affects the phenotype of the other homozygote.
Written as a lower case letter.";
        questionsArray[42] = "The allele that affects the phenotypes of one of the homozygotes and the
heterozygote. Written as a capital letter";
        questionsArray[43] = "A strong bond between two atoms formed by the sharing of valence
electrons";
        questionsArray[44] = "A chemical reaction that involves the breaking of a bond by the addition
of a water molecule. The reverse of a condensation reaction.";



    }

    public String getQuestion(int i)
    {
        return questionsArray[i];
    }


    public  String getAnswer(int i)
    {
        return answersArray[i];
    }
    public int getArrayLength()
    {
        return arrayLength;
    }

    // Only allows alteration of each corresponding question and answer at a time, to prevent a loss of
their correspondence.
    public  void setQuestionAndAnswer(int i,String question, String answer)
    {
        i = i - 1;
        questionsArray[i] = question;
        answersArray[i] = answer;
    }
    // Retrieves and stores a random question and its corresponding answer.
    public void arrayLength()
    {
        arrayLength = questionsArray.length;
    }
    // Prints the contents of both arrays.
```

```
   public void printArray()
   {
      int count = 0;
      System.out.println("\f");
      while (count < arrayLength - 1) {
         int questionNumber = count + 1;
         System.out.println(questionNumber + ". " + questionsArray[count] + " The answer is: " +
answersArray[count]);
         count++;
      }
   }

}
```

File Data Class

```
import java.io.IOException;

public class FileData
{
   private String file_name;

   public void FileData(String[] args) throws IOException {
      file_name = "QuizScores.txt";
   }
   public void readFile () throws IOException {
      try {
         ReadFile file = new ReadFile(file_name);
         String[] lines = file.OpenFile();
         int i;
         System.out.println(lines[0]) ;
         for ( i=1; i < lines.length; i++ ) {
            {
            System.out.println("attempt " + i + ": " + lines[ i ] ) ;
         }
         }
      }
      catch (IOException e) {
         System.out.println(e.getMessage());
      }
   }
   public void writeToFile(String textLine) throws IOException{
      try {
      WriteFile data = new WriteFile("QuizScores.txt.", true);
      data.writeToFile(textLine);
      System.out.println( "Text file written to." );
   }
   catch (IOException f) {
      System.out.println(f.getMessage());
   }
   }
```

```
}
```

Read File Class

```
import java.io.IOException;
import java.io.FileReader;
import java.io.BufferedReader;

public class ReadFile
{
  private String path;

   /**
    * Constructor for objects of class ReadFile
    */
   public ReadFile(String file_path)
   {
      path = file_path;
   }

   public String[] OpenFile() throws IOException {
      FileReader fr = new FileReader("QuizScores.txt");
      BufferedReader textReader = new BufferedReader(fr);
      int numberOfLines = readLines();
      String[] textData = new String[numberOfLines];
      int i;
      for (i=0; i < numberOfLines; i++) {
            textData[i] = textReader.readLine();
         }
     textReader.close();
     return textData;
   }

   public int readLines() throws IOException {
      FileReader file_to_read = new FileReader("QuizScores.txt");
      BufferedReader bf = new BufferedReader(file_to_read);
      String aLine;
      int numberOfLines = 0;
      while ((aLine = bf.readLine()) != null) {
         numberOfLines++;
      }
      bf.close();
      return numberOfLines;
   }
}
```

Write to file

```
import java.io.FileWriter;
import java.io.PrintWriter;
```

```
import java.io.IOException;


public class WriteFile
{
   private String path;
   private boolean append_to_file = false;
   /**
    * Constructor for objects of class WriteFile
    */
   public WriteFile(String file_path)
   {
      path = file_path;
   }

   public WriteFile( String file_path , boolean append_value ) {
      path = file_path;
      append_to_file = append_value;
   }

   public void writeToFile(String textLine) throws IOException {
      FileWriter write = new FileWriter(path, append_to_file);
      PrintWriter print_line = new PrintWriter( write );
      print_line.printf("%s" + "%n", textLine);
      print_line.close();
   }


}
```