

General Certificate of Education  
Advanced Subsidiary  
Specimen Pre-release material for 2009 and 2010  
Month Year



## COMPUTING

### COMP1/PR Problem Solving, Programming, Data Representation and Practical Exercise

Date Time

Pre-release material – Instructions for Candidates: Program Language Java

**To be given to candidates on or after 1 April XXXX**

#### Information

- There are 2 parts to this pre-release material:
  - Section A: Skeleton Program
  - Section B: Test Data.
- This material will be re-printed in the examination paper.
- You are advised to familiarise yourself with this material before the examination.
- You will use both the program and test data in the examination and your teacher will provide you with access to these electronically at the start of the examination.
- You can view/download this material on the AQA Website: [www.aqa.org.uk/xxxxxx](http://www.aqa.org.uk/xxxxxx)

**COMP1/PR/J**

---

## SECTION A – Skeleton Program<sup>1</sup>

---

Program statements

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;

////////////////////////////////////
////////////////////////////////////
/**
 * Written by Principal Examiner
 *
 * completed by .....
 * centre number .....
 * candidate number .....
 *
 * some statements are incomplete and
 * therefore do not do what they should do
 *
 * This program is designed to accept
 * a user's menu choice to:
 * 1. read in a single hexadecimal number
 *   and display the binary equivalent
 * 2. read in a text file of hexadecimal numbers and
 *   produce a text file of their binary equivalent
 * 3. display the content of a text file
 * 4. finish running the program
 */
public class SkeletonProgramSpecimen {

    Console console = new Console(); //An AQA class for I/O to console
    int choice;
    boolean finished = false;

    private void showMenu() {
```

---

<sup>1</sup> The Java solution has been created using Netbeans 6.1.

It has been attempted to stick to the Pascal skeleton given elsewhere on this page of the AQA Webiste, but there are some differences, which are mentioned where they exist. The two main ones are:

- Parameters – where Java only has pass by value. Primitive data types are obviously 'by value' but parameters pointing at class instances can change the instance, but the parameter can be changed without affecting the class instance.
- Regular expressions – the easiest way seemed to be to convert the char to a String, which then neatly demonstrated the standard use of regex; Strings cannot be used in a switch statement so the original char is used there.

---

```
console.println(" Please choose an option\n");
console.println(" 1 - Convert a Hex number to Binary");
console.println(" 2 - Convert a text file of hex numbers");
console.println(" 3 - Display text file");
console.println(" 4 - Exit the program\n");
} // end showMenu

//The original version uses call by reference, which Java does not use
//so it has been changed here to a method returning a value (a function)
private int getResponse() {
    String response = console.readLine("Enter option number: ");
    return -1; // a dummy response to keep Java happy
} // end getResponse

//isHexDigit is used in method binary to check whether a character is a
//hexadecimal digit.
//Strings have the easily readable regex facility,
//but they cannot be used in switch statements
private boolean isHexDigit(char c) {
    String s = String.valueOf(c);
    return s.matches("[0-9A-F]");
} // end isHexDigit

private String binary(String hex) {
    int noOfHexDigits;
    char thisHexDigit;
    String binaryEquivalent = "";
    String result = "";
    noOfHexDigits = hex.length();

    for (int hexDigit = 0; hexDigit < noOfHexDigits; hexDigit++) {
        thisHexDigit = hex.charAt(hexDigit);
        if (isHexDigit(thisHexDigit)) {
            switch (thisHexDigit) {
                case '0':
                    binaryEquivalent = "";
                    break;
                case '1':
                    binaryEquivalent = "";
                    break;
                case '2':
                    binaryEquivalent = "";
                    break;
                case '3':
                    binaryEquivalent = "";
                    break;
                case '4':
                    binaryEquivalent = "";
                    break;
                case '5':
                    binaryEquivalent = "";
                    break;
```

---

---

```
        case '6':
            binaryEquivalent = "";
            break;
        case '7':
            binaryEquivalent = "";
            break;
        case '8':
            binaryEquivalent = "";
            break;
        case '9':
            binaryEquivalent = "";
            break;
        case 'A':
            binaryEquivalent = "";
            break;
        case 'B':
            binaryEquivalent = "";
            break;
        case 'C':
            binaryEquivalent = "";
            break;
        case 'D':
            binaryEquivalent = "";
            break;
        case 'E':
            binaryEquivalent = "";
            break;
        case 'F':
            binaryEquivalent = "";
            break;
    } // end case
} else {
} // end if
result = result + binaryEquivalent;
} // end for loop
return result;
} // end binary

private void convertNumber() {
    String hexadecimalNumber, converted;
    hexadecimalNumber = console.readLine("Enter a hexadecimal number: ");
    converted = binary(hexadecimalNumber);
    console.println(converted);
} // end convertNumber

private void convertFile() {
    String fileNameIn = "hexFile.dat";
    String hexNumber, binaryNumber;
    try {
        BufferedReader hexFile = new BufferedReader(new FileReader(fileNameIn));

        while ((hexNumber = hexFile.readLine()) != null) {
```

---

---

```
        binaryNumber = binary(hexNumber);
        console.println(binaryNumber);
    } // end while
    hexFile.close();
} catch (IOException ioe) {
    console.println(ioe.toString());
    System.exit(0);
}
} // end convertFile

private void displayFile() {
    String fileNameIn = "binaryData.dat";
    String binaryLine;
    try {
        BufferedReader binaryFile = new BufferedReader(new FileReader(fileNameIn));
        while ((binaryLine = binaryFile.readLine()) != null) {
            for (int i = 0; i < binaryLine.length(); i++) {
                char bit = binaryLine.charAt(i);

                console.print(bit);
            } //end for
            console.println();
        } // end while
        binaryFile.close();
    } catch (IOException e) {
        console.println(e.toString());
        System.exit(0);
    } // end try/catch
} // end displayFile

public void writeToFile(String fileName, String b) {
    try {
        FileWriter outFile = new FileWriter(fileName);
        PrintWriter out = new PrintWriter(outFile);
        // Write text to file
        out.println(b);
        out.close();
    } catch (IOException e) {
        e.printStackTrace();
    } // end try/catch
} // end writeToFile

private void getChoices() {
    finished = false;

    showMenu();
    choice = getResponse();
    switch (choice) {
        case 1:
            convertNumber();
            break;
        case 2:
```

---

---

```
        convertFile();
        break;
    case 3:
        displayFile();
        break;
    case 4:

        break; // end program
    } // end case

} // end getChoices

public SkeletonProgramSpecimen() {
    getChoices();
} // end constructor SkeletonProgramSpecimen

public static void main(String[] args) {
    new SkeletonProgramSpecimen();
    System.exit(0);
} // end main
} // end class SkeletonProgramSpecimen

/* Class for use in AQA COMP1 exams for students using Java.
 *
 * class Console will be made available for students to use,
 * although they are not restricted to just using it for i/o to the console.
 * If they know how to use java.io.Console that is fine.
 * Also, System.out.println can be used for outputs.
 * This class should save time in the exam, especially if they
 * have been used by students beforehand.
 */
class Console {

    public Console() {
    } // end of constructor CommandLineInput
    //returns the first character of a line typed in.

    public char readChar() {
        return readChar("");
    } // end method readChar

    public char readChar(String prompt) {
        return readLine(prompt).charAt(0);
    } // end method readChar

    public byte readByte(String prompt) {
        try {
            return Byte.parseByte(readLine(prompt).substring(0, 1));
        } catch (NumberFormatException nfe) {
            println(nfe.toString() + "Parsing a Byte");
        } catch (Exception e) {
```

---

---

```
        println("another exception" + e.toString());
    }
    return -1;
} // end method readByte

public byte readByte() {
    return readByte("");
} // end method readByte

public String readLine() {
    return readLine("");
} // end method readLine with no parameter

public String readLine(String prompt) {
    String input = "";
    System.out.print(prompt);
    InputStreamReader isr = new InputStreamReader(System.in);
    BufferedReader br = new BufferedReader(isr);
    try {
        input = br.readLine();
    } catch (IOException ioe) {
        println("IO Error reading from command line.");
        System.exit(1);
    }
    return input;
} // end method readLine

public void print(Object o) {
    String output = String.valueOf(o);
    System.out.print(output);
} // end method print

public void println() {
    System.out.println();
} // end method println with no parameter

public void println(Object o) {
    String output = String.valueOf(o);
    System.out.println(output);
} // end method println

public void write(Object o) {
    print(o);
} // end method write

public void writeLine() {
    println();
} // end method writeLine with no parameter

public void writeLine(Object o) {
    println(o);
} // end method writeLine
```

---

---

```
public void printf(String format, Object args) {  
    System.out.printf(format, args);  
} // end method printf  
} // end class Console
```

---



---

**SECTION B – Test Data**

---

000000000000  
000001110000  
000010000100  
000010001000  
000001110000  
001000100010  
000100100100  
000010101000  
000001110000  
000000100000  
000000100000  
000000100000  
000001010000  
000010001000  
000100000100

**END OF PRE-RELEASE MATERIAL**