```java
package xmlfile;

import java.io.*;
import java.util.ArrayList;
import java.util.*;

/**
 * <p>Title: XML modelling of cogwheel phase cycling MQMAS sequence</p>
 *
 * <p>Description: The resulting XML document, called mqmas.xml, models
 * the three-pulse MQMAS sequence applied to a half-integer spin system.
 * pAB is the maximum value for the coherence order between pulses A and B.
 * pBC is the maximum value for the coherence order between pulses B and C.
 * The number of steps for the cogwheel phase cycling is N.
 * The winding number (windingA or wA) of pulse A must be different of zero.
 * Those of pulses B (windingB or wB) and C (windingC or wC) are constants.
 * </p>
 *
 * <p>Copyright: Copyright (c) 2007</p>
 *
 * <p>Company: pascal-man.com</p>
 *
 * @author not attributable
 * @version 1.0
 */

public class XmlDataCogwheelwA {

    private ArrayList notes;

    public XmlDataCogwheelwA() {
        notes = new ArrayList();
    }

   /**
     * The string of addElem() contains: element
     * example1: <element>
     * example2: </element>
     * @param element String
     * @return String
     */
    public String addElem (String element) {
        return "<" + element + ">" + "\n";
    }

    /**
     * The arrayList of addElemAttriArray() contains: element, attribute1,
     * attribute1 value, attribute2, attribute2 value, ...
     * example: <element attribute1='attribute1 value' ...>
     * @param element ArrayList
     * @return String
     */
    public String addElemAttriArray (ArrayList element) {
        String data = new String();

        data = "<" + element.get(0);
        for (int i=1; i<element.size(); i+=2) {
            data += " " + element.get(i) + "='" + element.get(i+1) + "'";
        }
        data += ">" + "\n";
        return data;
    }
    /**
     * The arrayList of addElemAttriEndArray() contains: element, attribute1,
     * attribute1 value, attribute2, attribute2 value, ...
     * example: <element attribute1='attribute1 value' ... />
     * @param element ArrayList
     * @return String
     */
    public String addElemAttriEndArray (ArrayList element) {
        String data = new String();

        data = "<" + element.get(0);
        for (int i=1; i<element.size(); i+=2) {
            data += " " + element.get(i) + "='" + element.get(i+1) + "'";
        }
        data += " />" + "\n";
        return data;
    }

    /**
     * This zeroArrayList() method zeroes the ArrayList element
     * @param element ArrayList
     */
    public void zeroArrayList (ArrayList element) {
        int n = element.size();
        while (n > 0)  {
            element.remove(n - 1);
            n -= 1;
        }
    }

    /**
     *
     * @param args String[]
     */
    public static void main(String[] args) {

        XmlDataCogwheelwA z = new XmlDataCogwheelwA();
        String elem = new String();
        int pAB = 3; //max coherence order p between 1st-pulse A and 2nd-pulse B
        int pBC = 3; //max coherence order p between 2nd-pulse B and 3rd-pulse C
        int N = 23;                   //winding number wA of the 1st-pulse A = N
        int windingB = 0;                 //winding number wB of the 2nb-pulse B
        int windingC = 1;                 //winding number wC of the 3rd-pulse C

        try {
            // XML file name for saving data in hard disk
            FileWriter fw = new FileWriter("mqmas.xml");

            //root of XML document
            elem = z.addElem("?xml version='1.0' encoding='ISO-8859-1' ?");
            fw.write(elem);
            elem = z.addElem("?xml-stylesheet type='text/xsl' href='mqmas.xsl' ?");
            fw.write(elem);

            //root of elements
            elem = z.addElem("mqmas");
            fw.write(elem);

            //first pulse A-----------------------------------------------------
            for (int i1 = -pAB; i1 <= pAB; i1++) {
                for (int j1 = 1; j1 < N; j1++) {
                    z.notes.add("A");                    //A for the first pulse
                    z.notes.add("p");                    //p for coherence order
                    z.notes.add(String.valueOf(i1));                       //pAB
                    z.notes.add("w");                     //w for winding number
                    z.notes.add(String.valueOf(j1));                        //wA
                    elem = z.addElemAttriArray(z.notes);
                    z.zeroArrayList(z.notes);
                    fw.write(elem);

                    //second pulse B============================================
                    for (int i2 = -pBC; i2 <= pBC; i2++) {
                        z.notes.add("B");               //B for the second pulse
                        z.notes.add("p");                //p for coherence order
                        z.notes.add(String.valueOf(i2));                   //pBC
                        z.notes.add("w");                 //w for winding number
                        z.notes.add(String.valueOf(windingB));              //wB
                        elem = z.addElemAttriArray(z.notes);
                        z.zeroArrayList(z.notes);
                        fw.write(elem);

                        //third pulse C, coherence order p = -1 ++++++++++++++++
                        z.notes.add("C");                //C for the third pulse
                        z.notes.add("w");                 //w for winding number
                        z.notes.add(String.valueOf(windingC));              //wC
                        elem = z.addElemAttriEndArray(z.notes);
                        z.zeroArrayList(z.notes);
                        fw.write(elem);
                        //end of third pulse C++++++++++++++++++++++++++++++++++

                        elem = z.addElem("/B");
                        fw.write(elem);
                    }
                    //end of second pulse B=====================================

                    elem = z.addElem("/A");
                    fw.write(elem);
                }
            }
            //end of first pulse A----------------------------------------------

            elem = z.addElem("/mqmas");
            fw.write(elem);
            fw.close();
        }
        catch(IOException ex) {
              ex.printStackTrace();
        }
    }
}
```