Work/Studies History

1. What was your emphasis in your bachelor's work at XXX?
2. What was the most interesting project you worked on there?
3. What is your emphasis in your master's work here at UF?
4. What is the most interesting project you worked on here?
5. Describe your work at XXX?
6. How was your code overseen? Team programming? Managerial oversight?

Programming

7. What is your level of familiarity with….
    a. C#
    b. Java
    c. ASP.net
    d. Visual Studio
8. Do you generally do programming in an IDE? Which ones?
9. Exercise 1
10. Exercise 2

XML / XSD

11. What is your level of familiarity with….
    a. XML
    b. DTD
    c. XSD
12. Exercise 3

Database

13. What is your level of familiarity with….
    a. Databases
    b. SQL
    c. Stored Procedures
14. What are some of the reasons to use stored procedures
15. Exercise 4
16. Triple-store databases / RDF / subject-predicate-object

Notes:

The purpose of this exercise is to have the student exhibit basic knowledge of OOP programming and C#.

The first page is provided to the user and they are asked to improve the code, particularly the ComplexNumber class.

Then, the updated class is given to them. While going over this code, the following questions are asked:

1) What does the `IEquatable<ComplexNumber>` part mean?
    a. What principal of OOP does this represent? (Inheritance)
    b. What is the difference between an interface and an abstract class
    c. What do the brackets represent? Experience with generics?
2) The two variables (`real_part` and `imaginary_part`) are now private.
    a. What principal of OOP does this represent? (Encapsulation)
    b. What if I had made them `protected`?
3) What is the `public ComplexNumber()` part called?
    a. There are two signatures. What principal of OOP does this represent? (Polymorphism)
4) The first Add method now hides the method of addition from the user.
    a. What principal of OOP does this represent? (Abstraction).
5) The second Add class is `static`.
    a. What does this mean?
    b. How would you call this method?
    c. What two things are wrong with the implementation of this method?
    d. What if I change this from a `class` to a `struct`?
    e. by reference versus by value
6) The `ToString()` method includes the word `override`.
    a. What does this mean?
    b. What class's method is it overriding from?
    c. What class does this class extend?

The code and output at the end of this exercise are used to show one of the problems with the implementation of the static method, since it actually alters the contents of the first variable. This is used to begin a converation about structs vs. classes and by reference versus by value parameters passing.

Exercise 1)

The ComplexNumber code below exhibits several poor coding practices.

How could this code be improved?

```
static class Program
{
    [STAThread]
    static void Main()
    {
        ComplexNumber first = new ComplexNumber();
        first.real_part = 4.1;
        first.imaginary_part = -2.1;

        ComplexNumber second = new ComplexNumber();
        second.real_part = -3.1;
        second.imaginary_part = 4.3;

        ComplexNumber sum = new ComplexNumber();
        sum.real_part = first.real_part + second.real_part;
        sum.imaginary_part = first.imaginary_part + second.imaginary_part;

        Console.WriteLine("The imaginary number is " + sum.Real_Part +
        "+" + sum.Imaginary_Part + "j");
    }
}

public class ComplexNumber
{
    public double real_part;
    public double imaginary_part;
}
```

```
public class ComplexNumber : IEquatable<ComplexNumber>
{
    private double real_part;
    private double imaginary_part;

    public ComplexNumber()
    {
        real_part = 0;
        imaginary_part = 0;
    }

    public ComplexNumber(double Real_Part, double Imaginary_Part)
    {
        real_part = Real_Part;
        imaginary_part = Imaginary_Part;
    }

    public double Real_Part
    {
        get { return real_part;  }
        set { real_part = value; }
    }

    public double Imaginary_Part
    {
        get { return imaginary_part; }
        set { imaginary_part = value; }
    }

    public void Add(ComplexNumber other)
    {
        real_part = real_part + other.Real_Part;
        imaginary_part = imaginary_part + other.Imaginary_Part;
    }

    public static ComplexNumber Add(ComplexNumber first, ComplexNumber second)
    {
        return first.Add(second);
    }

    public override string ToString()
    {
        if (imaginary_part != 0)
        {
            if (imaginary_part < 0)
                return real_part.ToString() + imaginary_part.ToString() +"j";
            else
                return real_part.ToString()+"+"+imaginary_part.ToString()+"j";
        }
        else
            return real_part.ToString();
    }

    public bool Equals(ComplexNumber other)
    {
        if ((real_part == other.Real_Part) &&
           (imaginary_part == other.Imaginary_Part))
        {
```

```
            return true;
        }
        else
        {
            return false;
        }
    }
}
```

```
static class Program
{
    [STAThread]
    static void Main()
    {
        ComplexNumber first = new ComplexNumber( 4.1, -2.1 );
        ComplexNumber second = new ComplexNumber( -3.1, 4.3 );

        Console.WriteLine("First imaginary number is " + first);
        Console.WriteLine("Second imaginary number is " + second);
        Console.WriteLine("The sum is " + ComplexNumber.Add( first, second));
        Console.WriteLine();
        Console.WriteLine("First imaginary number is " + first);
        Console.WriteLine("Second imaginary number is " + second);
    }
}
```

OUTPUT FROM ABOVE)

```
First imaginary number is 4.1-2.1j
Second imaginary number is -3.1+4.3j

The sum is 1+2.2j

First imaginary number was 1+2.2j
Second imaginary number was -3.1+4.3j
```

OUTPUT FROM STRUCT VERSION)

```
First imaginary number is 4.1-2.1j
Second imaginary number is -3.1+4.3j
The sum is 1+2.2j

First imaginary number was 4.1-2.1j
Second imaginary number was -3.1+4.3j
```

Notes:

The purpose of this exercise is to have the student walk through a fairly complicated problem and determine a solution. The application is not required to write a solution, only to work out how the solution would be implemented.

The application is asked to NOT use the type in building the tree, just use the second table of relationships.

Exercise 2)

Given the two data tables below, how would you take this data and create the resulting tree view?

ID	Name	Type
1	United States	Country
2	Florida	State
3	Duval	County
4	Jacksonville	City
5	Alachua	County
6	Alachua	City
7	Gainesville	City
8	Bradford	County
9	Starke	City
10	Georgia	State
11	District of Columbia	District
12	Washington	City

Parent ID	Child ID
1	2
1	10
1	11
2	3
2	5
2	8
3	4
5	6
5	7
8	9
11	12

1. United States (Country)
    a. District of Columbia (District)
        i. Washington (City)
    b. Florida (State)
        i. Alachua (County)
            1. Alachua (City)
            2. Gainesville (City)
        ii. Bradford (County)
            1. Starke (City)
        iii. Duval (County)
            1. Jacksonville (City)
    c. Georgia (State)

Notes:

The purpose of this exercise is to have the student exhibit a working knowledge of XML and XML Schemas.

First part)

1. Name is in the wrong order… <xs:sequence>
2. Max number of vets = 1
3. Kitten is not in the controlled list

Second part)

The resultant pets will only represent pets owned by ONE owner, since it has been moved out of the pet collection.

Third part)

```
<rolodex>
  <contact>
    <name>
      <first>Woody</first>
      <last>Guthrie</last>
    </name>
    <phone type="work">(352) 123-4567</phone>
    <phone type="cell">(352) 123-4567</phone>
    <email>Woody.Guthrie@gmail.com</email>
  </contact>
  <contact>
    <name>
      <first>Bob</first>
      <last>Dylan</last>
    </name>
    <phone type="work">(123) 456-7890</phone>
    <email type="personal">B.Dylan@gmail.com</email>
    <email type="business">SingerSongwriter@cnn.com</email>
  </contact>
</rolodex>
```

Exercise 3)

The XML below does not conform to the XSD on the next page. Identify the problem(s).

```
<pets>
  <pet>
    <name>Peanut</name>
    <name>Fluffy</name>
    <owner>Gail</owner>
    <veterinarian> Albert Einstein </veterinarian>
    <type>Dog</type>
  </pet>
  <pet>
    <name>Jaala</name>
    <owner>Mark</owner>
    <veterinarian>John Schultz</veterinarian>
    <veterinarian>Albert Einstein</veterinarian>
    <type>Kitten</type>
  </pet>
  <pet>
    <name>Isis</name>
    <owner>Mark</owner>
    <type>Kitten</type>
  </pet>
</pets>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="pets">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="pet">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="owner" type="xs:string" maxOccurs="1" />
              <xs:element name="veterinarian" type="xs:string" maxOccurs="1" />
              <xs:element name="name" type="xs:string" maxOccurs="unbounded" />
              <xs:element name="type" maxOccurs="1" >
                <xs:simpleType>
                  <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="Dog" />
                    <xs:enumeration value="Mouse" />
                    <xs:enumeration value="Cat" />
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

The schema below is different. How does the data it represents differ from the data in the first example?

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="pets">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="owner" type="xs:string" maxOccurs="1" />
        <xs:element maxOccurs="unbounded" name="pet">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="veterinarian" type="xs:string" maxOccurs="1" />
              <xs:element name="name" type="xs:string" maxOccurs="unbounded" />
              <xs:element name="type" maxOccurs="1" >
                <xs:simpleType>
                  <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="Dog" />
                    <xs:enumeration value="Mouse" />
                    <xs:enumeration value="Cat" />
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Write XML for the following two contacts using the schema below

Woody Guthrie
(352) 123-4567 (work)
507-123-45678 (cell)
Woody.Guthrie@gmail.com

Bob Dylan
(123) 456-7890 (work)
B.Dylan@gmail.com (personal)
SingerSongwriter@cnn.com (work)

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="rolodex">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="contact">
          <xs:complexType>
            <xs:sequence>
              <xs:element minOccurs="1" maxOccurs="1" name="name" type="contactName" />
              <xs:element maxOccurs="unbounded" name="phone" type="contactPhone" />
              <xs:element maxOccurs="unbounded" name="email" type="contactEmail" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="contactName">
    <xs:sequence>
      <xs:element name="first" type="xs:string" />
      <xs:element name="last" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
```

```
  <xs:complexType name="contactPhone">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="type" type="phoneTypes" use="required" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:simpleType name="phoneTypes">
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="cell" />
      <xs:enumeration value="home" />
      <xs:enumeration value="work" />
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="contactEmail">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="type" type="emailTypes" use="optional" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:simpleType name="emailTypes">
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="personal" />
      <xs:enumeration value="business" />
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

Notes:

The purpose of this exercise is to have the student work through a growing database design. They should demonstrate agility while tackling the problem, as their first solution will almost certainly not allow all of the different data to be stored correctly. Slowly reveal to the applicant each piece of new data to be stored and let them tackle the database structure changes that would require.

Finally, discuss the EAC standard as a solution to this problem. So many little pieces of data.. too much to store it all in the database likely.

Database holds only those things necessary to find the person, etc..

Discuss triple-store etc.. as a search mechanism in the database to locate the entity to display.

Exercise 4)

Design a database to store the data found in the natural language below.

> Eric Schwartz is forty-two years old and has one son, Will Schwartz (6 years old) and two daughters, Jill (4 years old) and Sue (3 years old).
>
> Eric Scwartz is married to Lisa Schwartz (maiden name of Pullman).

The database should be be updated to reflect their occupations, etc..

Eric is a lecturer in the Art Department at UF and Lisa is an assistant professor in the Engineering Department at UF.

The database should be be updated to reflect this new piece of data:

```
Lisa is also the director of the Machine Intelligence Lab at UF.
```

The database should be be updated to reflect this new piece of data:

Eric was previously married to Nancy Martin (who is currently unemployed) and his son (Will) is actually Nancy’s son, not Lisa’s.

The database should be be updated to reflect this new piece of data:

```
From 2001 to 2005, Nancy and Lisa co-owned a business called 'Intelligence Unnatural, Inc'
which was based in Ocala, Florida.
```

The database should be be updated to reflect this new piece of data:

```
In 2009, Nancy passed away.
```

Part-time .NET Development Position / Digital Library Center

Need a part-time .NET developer for creation of the classes and editor for the open-source EAC metadata editor. Programming will be contributed to SobekCM, which powers UFDC and dLOC, and distributed openly to other interested groups. Pay will be $15/hour for 240 hours and all work should be completed within sixteen weeks.

Ideal candidates will have the following skills and experience:

- Proficiency with C#, SQL Server, and Transact-SQL
- Strong object oriented development
- Familiarity with XML, XSLT, and related technologies
- Good technical, analytical and problem solving skills required
- Self motivated with excellent verbal and written communication skills

Additional considerations for skills or experience in:

- User interface and graphics design
- WinForms development
- ASP.net development
- Experience writing and debugging SQL stored procedures

The successful candidate will be required to:

1. Take the EAC schema and create a class library with the interfaces and classes to hold the data. The final classes will be *Serializable* and implement the *iEquatable* interface, as appropriate for each object. Collections of objects will be stored as generic collections, with public access via *ReadOnlyCollections* and appropriate gettors and settors (i.e., 'Add', 'Remove'). Additional internal methods will allow for clearing the collections and comparing/merging collections.
2. Create a static reader class which will accept either a file name or a *Stream* object, convert to a *XmlTextReader*, and then iterate through each node, fully populating the data structure
3. Create a static writer class which will write EAC-compliant XML to a *Stream* object, using a minimal number of *StringBuilder* intermediate objects, as necessary.
4. Design database tables and stored procedures to store the main EAC information which will be need to be retained in a pre-existing database.
5. Create the *iDatabase* interface, static database gateway class, and appropriate database connection classes for saving the pertinent data to the database and retrieving the pertinent data from the database.
6. [AS TIME PERMITS] Create windows forms and/or web forms to allow users to view and edit the information from the EAC files.

EAC Details:
The EAC schema is based on the Extensible Markup Language (XML), which enables the display, discovery and sharing of contextual information. The standard is maintained by the Society of American Archivists in partnership with the Berlin State Library, and is compatible with ISAAR(CPF), the International Standard Archival Authority Record for Corporate Bodies, Persons, and Families. EAC allows us to encode contextual information about the creation and use of historical records by agents including individuals, families, and organizations. In addition to basic biographical and historical data,

the contextual information may describe functions, activities, geographic places, events, and relationships to other agents.

The richness and flexibility of the EAC schema makes it possible to enhance all digital services based on provenance. EAC supports the linking of contextual information about record-creating agents to digital object metadata or to descriptions of library and archives holdings. It also supports the linking of contextual information about one agent to contextual information describing other agents, based on defined relationships between the agents and/or their records. EAC can be used for authority file encoding either as a standalone schema or in combination with other standards. For example, EAC can be used in conjunction with the Encoded Archival Description (EAD) schema for encoding and delivering archival finding aids. EAD would be used to encode descriptive information about archival records, and EAC would be used to encode contextual metadata about the agents responsible for creating the records.

Background on SobekCM (UFDC and dLOC)

Full proposal
http://www.uflib.ufl.edu/ufdc/?m=hdPD&b=UF00095574&v=00001