Redo Ques. 9.

CIS 400 Exam I

Name: Robert Clancy WID: 84266 4019 Section: B

Vocabulary

The following questions deal with object-orientation on the theoretical level, not a specific language

Q1. What is structured programming?

Structured programming is a style of writing code where the program's logic is entirely handled by control-flow statements and NOT goto statements. The end result is code that is much easier to read.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Q2. What does it mean to cast a variable?

casting a variable is simply changing the type of that variable. You can't cast "Hi" to an int, but you can cast "234" from an int to a string. Every language supports different rules for implicit/explicit casting.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Q3. Describe the relationship between a base class and derived class:

In general, a base class has multiple derived classes. A base class called Pencil might have attributes like color, width, length, and weight and the derived classes called no1pencil and no2pencil would inherit Pencil's attributes, then have their own attributes or methods.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

C# Keywords

The following questions deal specifically with the C# language

Q4. What does the keyword "protected" mean when used with a method?

protected methods cannot be called within any other class except the class that defined the protected method and that class's child classes.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Q5. What does the "override" keyword mean when used with a method?

a method with an accessor named override is telling the compiler that there is another method out there with the same name and to ignore that method and instead call the override method.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Q6. What does the "sealed" keyword mean when used with a class?

Sealed classes cannot be inherited from.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Coding Questions

The following questions deal with the UML Diagram on the last page of the Exam:

Q7. What is the relationship between RiftenRippleChips, Side, and IMenuItem?

RiftenRuffleChips is a child class of Side. Side implements the IMenuItem interface. Because Side implements IMenuItem, all of its child classes (like RiftenRuffleChips) also implement IMenuItem.

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Q8. Write the code corresponding to the DonutTopping enum:

```
public enum DonutTopping {
    Chocolate,
    Sprinkles,
    Bacon
}
```

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Q9. Complete the tests of RiftenRippleChips:

```
[Theory]
[InlineData(Size.Small, 1.45)]
[InlineData(Size.Medium, 2.23)]
[InlineData(Size.Large, 3.41)]
public void ShouldHaveCorrectPriceForSize(Size size, Price price)
{
```

```
RiftenRippleChips rrc = new RiftenRippleChips();
rrc.Size = size;
Assert.Equal(price, rrc.size);
```

```
}
```

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

```
[Fact]
public void ShouldDefaultToSmall()
{
```

```
RiftenRippleChips rrc = new RiftenRippleChips();
Assert.Equal(Size.Small, rrc.Size);
```

```
}
```

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

```
[Theory]
[InlineData(Size.Small)]
[InlineData(Size.Medium)]
[InlineData(Size.Large)]
public void ShouldNotHaveAnySpecialInstructionsRegardlessOfSize(Size size)
{
```

```
RiftenRippleChips rrc = new RiftenRippleChips();
rrc.Size = size;
Assert.Empty(rrc.SpecialInstructions);
```

```
}
```

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Redo

Q10. Write an implementation of the DawnstarDonught, assuming the price is $3.50, the calories are 451, there are never any special instructions, and the ToString() method returns “Dawnstar Donut with Chocolate”, “Dawnstar Donut with Sprinkles”, or “Dawnstar Donut with Bacon” depending on the topping selected:

```
public class DawnstarDonught : IMenuItem: {
    public override double Price {
        get { return 3.50 }
    }
    public override int Calories {
        get { return 451 }
    }
    public DonutTopping DonutTopping { get; set }
    public override List<string> SpecialInstructions {
        get {
            List<string> specialList = new List();
        }
    }
    public override ToString() {
        return($"Dawnstar Donut with {DonutTopp}");
    }
```

Confidence (circle one): very confident - (5) (4) (3) (2) (1) - Not confident at all

Coding Diagrams

This UML Diagram is used with questions 7 through 10.